Journal of Big Data

METHODOLOGY Open Access

Traffic flow estimation with data from a video surveillance camera

Aleksandr Fedorov[1*], Kseniia Nikolskaia[1], Sergey Ivanov[1], Vladimir Shepelev[1] and Alexey Minbaleev[2]

*Correspondence:
canxes@mail.ru
[1] South Ural State University, 454091 Chelyabinsk, Russia
Full list of author information is available at the end of the article

Abstract

This study addresses the problem of traffic flow estimation based on the data from a video surveillance camera. Target problem here is formulated as counting and classifying vehicles by their driving direction. This subject area is in early development, and the focus of this work is only one of the busiest crossroads in city Chelyabinsk, Russia. To solve the posed problem, we employed the state-of-the-art Faster R-CNN two-stage detector together with SORT tracker. A simple regions-based heuristic algorithm was used to classify vehicles movement direction. The baseline performance of the Faster R-CNN was enhanced by several modifications: focal loss, adaptive feature pooling, additional mask branch, and anchors optimization. To train and evaluate detector, we gathered 982 video frames with more than 60,000 objects presented in various conditions. The experimental results show that the proposed system can count vehicles and classify their driving direction during weekday rush hours with mean absolute percentage error that is less than 10%. The dataset presented here might be further used by other researches as a challenging test or additional training data.

Keywords: Traffic flow estimation, Traffic analysis, Vehicle detection, Convolutional neural network, Surveillance data

Introduction

Urbanization and increased building density of cities are essential features of modern society. Not only does such a way of life bring economic benefits, but it also poses a new set of problems for city authorities. One of these problems is efficient traffic management and analysis. High population density leads to the tremendous number of personal cars, an increased number of freight vehicles for transportation of commodities and goods, tight pedestrian traffic. Transportation tasks can no longer be addressed by sub-optimal heuristics, based on the small amount of the manually gathered statistics. To make efficient decisions, forecast and assess their consequences, authorities require an automated system for analyzing traffic flow throughout the city.

Nowadays, many cities have low-cost video surveillance systems, also known as closed-circuit television (CCTV). They exhibit rapid growth nowadays and usually include heterogeneous cameras with various resolution, mounting points, and frame rates [43]. CCTV works 24 h a day, 7 days a week and generates a massive amount of information, called Big Data. Among other applications, this data can serve as a foundation for the automated traffic surveillance system.

There are several well-known problems when working with Big Data. For instance, to build a traffic surveillance system, one has to develop efficient algorithms for moving, storing, intelligent processing and analyzing data from surveillance cameras. In this work, we focus on the last two aspects.

Despite obvious benefits, there is a limited number of researches that aim to analyze real-world data from the CCTV systems. Some of the representative works are the studies [42] and [24], which utilize low-resolution heterogeneous CCTV data and deep neural networks to count vehicles on the road and estimate traffic density. The examples of using conventional computer vision techniques are the systems developed in works [25, 34]. They address the problems of freight traffic monitoring [34] and illegal traffic behaviour recognition [25], respectively.

In comparison, great efforts were made to improve results in more specific tasks. For vehicle detection, the majority of modern works focuses on adapting and improving state-of-the-art object detection frameworks such as Faster R-CNN [28], YOLO [26], and SSD [21]. This includes architectural novelties [10, 12, 15, 32, 36, 44], exploiting temporal information to perform joint detection and tracking [14, 16, 22], improving detection speed [1]. Another example of the well-studied task in intelligent transportation is fine-grained vehicle classification [31, 33].

In this work, we aim to address a gap between recent researches and them real-world applications. As a case study, we focused on one of the busiest crossroads in Chelyabinsk city, Russia. We started from a widely adopted Faster R-CNN model and sequentially improved baseline performance via additional mask branch [9], anchors shape optimization, focal loss [37] and adaptive feature pooling [20]. The proposed model, together with simple and efficient SORT tracker [2], forms a backbone of the modern traffic analysis solution. Our system is capable of counting and analyzing vehicles movement direction with a maximum relative error that is lower than 10%. The results of this research will be used by city authorities to improve the overall throughput of the crossroad.

The contribution of the paper can be summarized as follows:

- We have constructed a new challenging dataset, which includes 982 highly crowded frames with more than 60,000 instances in total.
- We have proposed and implemented a novel traffic flow estimation system, which is based on the recent advances in vehicle detection and tracking tasks.
- We thoroughly evaluated our system and provided empirical evidence that the proposed solution has sufficient precision and might be further used as a backbone for other high-level models.

The paper has the following structure. A review of relevant studies is presented in "Related works" section. "Methodology" section describes our approach to the traffic flow estimation problem, discusses design choices, and collected dataset. Evaluation protocol and experiment implementation details are presented in the "Experiments" section. In "Results and discussions" section, we provide analysis and summary of the obtained results. Finally, "Conclusions" section concludes the paper and outlines directions for further research.

Related works

There are a very limited number of researches in the area of analyzing data from video surveillance systems. The most representative works are briefly summarized in the introduction. However, the targets and goals of these studies are different from the scope of this paper.

For the lack of alternatives, here we provide an overview of the recent advances in the research topics on which our work is based on, i.e. objects detection and multi-object tracking (MOT) problems. When possible, we additionally highlight advances in specialized works for vehicle detection and tracking.

Objects detection

As of today, most of the object detectors are based on the convolutional neural networks (CNN) and can be roughly divided into two classes: single-stage detectors and two-stage detectors. The single-stage detectors are generally fast and predict objects bounding boxes together with classes within a single network pass. Classical examples of the single-stage detectors are YOLO [26] and SSD [21]. These architectures work particularly well in cases when target objects occupy a considerable amount of image. An example of such data is the popular UA-DETRAC vehicle detection dataset [39]. Based on this data, Dmitriy Anisimov and Tatiana Khanova [1] have shown that a thoroughly constructed SSD-like detector can run faster than 40 frames per second on the modern CPU while maintaining favourable precision. Another example of good speed-precision trade-off is YOLO v2 architecture [30], which was specialized for vehicle detection via anchors clustering, additional loss normalization, and multi-layer feature fusion strategy.

The most representative example of two-stage detectors is the R-CNN family of detectors [7–9, 28] that currently occupy leading places in the COCO [19] and Cityscapes [5] benchmark datasets. In comparison to single-stage detectors, two-stage detectors first predict regions and then refine and classify each of them during the second stage. Early R-CNN [8] work utilized a straightforward approach: regions were generated via a selective search algorithm and then fed to the classification CNN. The overall speed of the R-CNN is low due to the selective search compute time and requirement to run heavy classifier per each region. To overcame this limitation, Fast R-CNN was proposed [7]. Instead of running CNN per each region, Ross Girshick fed the whole image to the CNN and pooled regions of interest (ROI) from the last feature map. Replacing selective search in the Faster R-CNN [28] with tiny CNN, called region proposal network, further boosted precision and speed of the detector.

A thorough review of the main advantages and drawbacks of single and two-stage detectors is presented in the work [11].

Many vehicle detection works adopted variants of the Faster R-CNN architecture. Li [16] proposed to process multiple adjacent frames to better handle blurring and short-term occlusions. Wang et al. [37] studied the application of the focal loss [18] for vehicle surveillance. They showed that being a relatively simple technique, focal loss provides substantial performance improvements. Hu et al. [10] focused on the improving scale robustness of the Faster R-CNN and suggested context-aware RoI (CARoI) pooling that utilizes deconvolution with bilinear kernels to represent features for small objects accurately. CARoI pooling works on top of the multiple layers and fuses high and low-level semantic information to improve performance.

Multi-object tracking

The progress in the precision of the object detectors mentioned above made tracking-by-detection paradigm the leading one in a multi-object tracking (MOT) task. In this paradigm, tracking is formulated as a data association (DA) problem where the aim is to merge fuzzy detections across multiple frames into long tracklets.

Classical tracking-by-detection methods rely solely on motion clues from the detector and solve the DA problem via optimization techniques. Well known examples are Multiple Hypothesis Tracking (MHT) [13] and Joint Probalistic Data Association Filter (JPDAF) [29]. These methods tackle the association problem on a frame-by-frame basis, yet their combinatorial complexity is exponential in the number of tracked objects, which makes them unsuitable for real-time tracking. On the contrary, recent SORT tracker [2] showed that a simple Hungarian algorithm with Kalman filtering for movement forecasting could achieve real-time processing speed while maintaining favourable performance.

Most of the recent improvements in the MOT task involve fusing motion features with appearance one to better distinguish highly occluded objects and reidentify lost instances. The appearance clues usually come from convolutional neural networks [4, 40]. However, Tang et al. [35] showed that hand-crafted features, like the histogram of oriented gradients and colour histograms, might also be used if no training data is provided. From the practical point of view, computing visual features for each tracked object leads to a highly increased computational burden, especially if the number of objects is high. Together with detector processing time, cumulative performance usually is far from being near real-time.

It is worthwhile to mention that simultaneous vehicle detection and tracking is an area of active research [14, 22]. The coupling of these tasks might solve the performance problem mentioned above. Detectors already incorporate appearance features, detection precision will also benefit from the temporal image context.

Methodology

This paper aims to develop a system for traffic flow estimation, i.e. for counting and classifying vehicles by their movement directions. To achieve that goal, we divide the problem into three sub-tasks: vehicle detection, vehicle tracking, and vehicle direction estimation. It naturally leads to a modular and easy to test architecture composed of the detection and tracking modules, respectively. In the following sections, we describe each module in detail alongside data collected for training and evaluation.

Dataset

There are more than 40 surveillance cameras in Chelyabinsk, the majority of each is mounted high above the main crossroads to provide a general overview of the traffic situation. In comparison to the low-end CCTV [43] systems, these cameras provide stable 25 frames per second, maintaining 1920×1080 resolution. However, the video stream is not perfect due to compression artefacts, blurring, and hardware faults. As shown in Fig. 1, images from these cameras are different from data in the public datasets,

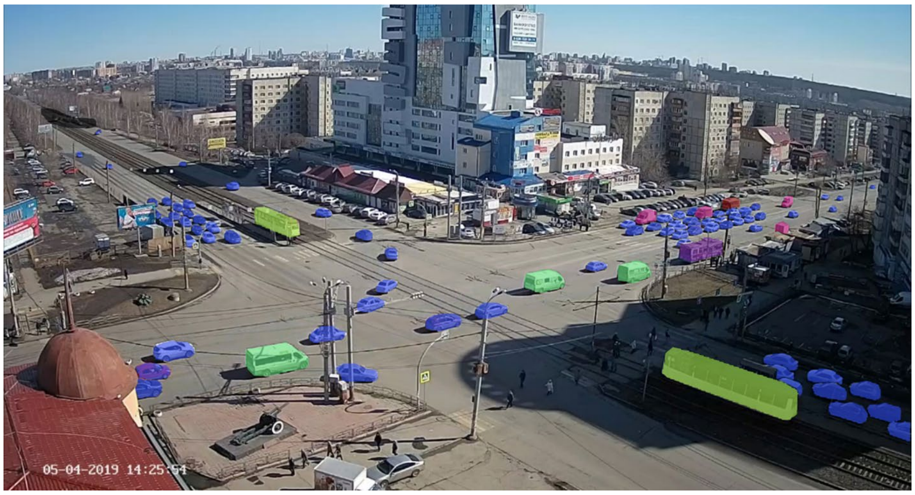

Fig. 1 Example of an image from our dataset. Blue, yellow, green, red, violet, pink and black polygons stands for the "car", "tram", "van", "bus", "trolleybus", "truck" and "unlabeled" objects respectively

like KITTI [6] or UA-DETRAC [39]. The difference comes from many aspects: viewing angle, number of instances on a single frame, occlusions level, high variations in scale. Our data is more challenging. Even a trained human eye is not capable of labelling every instance on the image from the first time.

For this work, we focused on a single camera that monitors one of the most problematic crossroads in Chelyabinsk city. It is a deliberate choice to maximize detection precision and obtain a minimum viable product. We annotated with polygons more than 60,000 instances across 982 frames from the selected camera. All annotation relevant tasks were performed in the COCO Annotator tool [3]. In comparison to the existent vehicle detection datasets [6, 39], we did not annotate video sequences, as it would be extremely time-consuming for such crowded scenes. Instead, we focused on covering a wide range of traffic situations, time of day, weather conditions. We also annotated every vehicle that can be distinguished with high confidence, especially in crowded traffic jams, as an approximate number of vehicles in a jam might be useful for real-world tasks in the future. The overall statistic about the gathered data is summarized in the Table 1. Despite the specificity of the proposed dataset, our data might be used in other studies as a supplementary data source or challenging test set.

Table 1 Data distribution in the collected dataset

Vehicle type	Total instances	Mean instances per image
Car	53083	54.06
Van	2783	2.83
Truck	2415	2.46
Tram	1298	1.32
Bus	1234	1.26
Trolleybus	611	0.62

To thoroughly validate the proposed system in the traffic flow estimation problem we collected statistics about passing vehicles from 8:00 to 12:00 and from 17:00 to 19:00 for four of the most popular movement directions. We have deliberately focused on morning and evening hours during the weekday when traffic flow is very dense, and tracking vehicles is a hard task even for humans.

Detection module

The foundation of our detection module is a widely adopted two-stage Faster R-CNN [28] detector. This choice was made for two reasons. First, two-stage detectors are current state-of-the-art in many detection benchmarks [5, 19, 39] and provide a solid start point in any detection task. Second, our future research may include complex multi-task learning. Extending Faster R-CNN architecture for multiple tasks is as simple as adding a new prediction branch.

We further improved Faster R-CNN baseline performance with a feature pyramid network (FPN) backbone [17], an additional mask branch [9], anchors shape optimization, focal loss [37] and adaptive feature pooling [20].

Mask branch

As was shown in the Mask R-CNN work [9], additional regression of the per-instance masks leads to better precision in the relevant bounding box regression task. Consequently, the first optimization that we applied was a complementary mask branch. This branch runs in parallel with the existing Faster R-CNN branches and aims to regress a binary mask for each region of interest. For simplicity, we followed [9] and approximated precise pixel-wise instance masks via coarse polygon masks from the collected dataset.

Anchors shape optimization

The next improvement comes from the simple observation that Faster R-CNN anchors parameterization was explicitly designed for the COCO dataset [9], but our target data is different. There is an effective way [27] to choose anchors by K-Means clustering with a distance measure that is based on the intersection over union (IOU). However, this procedure yields a heterogeneous set of anchors, which is not suitable for Faster R-CNN architecture, specifically for the region proposal network (RPN). RPN regresses offsets for anchors which are conventionally defined as a Cartesian product of scale and aspect ratio sets. Each anchor is assigned to the specific FPN level, based on its area. RPN is shared across multiple feature pyramid levels and therefore, for each level RPN predicts offsets for anchors that have the same scale but different aspect ratios.

We have found that in the case of using transfer learning any naive strategy of incorporating heterogeneous anchors into the described framework leads to unstable training and bad performance. For our setup, it holds even if separating region proposal network for each pyramid level. To adjust anchors to our data and keep benefits of the pre-trained weights, we employed a grid search. The objective of the search was to find a set of aspects and scales that maximizes IOU between objects in the training set and anchors that are matched to them during training. We followed initial Faster R-CNN design and limited the scope of the search for anchor scales exclusively to sequences of powers of two.

Focal loss

To address extreme foreground-background class imbalance encountered during the RPN training we used novel focal loss, originally proposed for single-stage detectors [18]. The focal loss (FL) serves as a drop-in replacement for conventional cross entropy and can be defined as:

$$p_t = \begin{cases} p & \text{if y = 1} \\ 1-p & \text{otherwise,} \end{cases}$$
$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma} * log(p_t)$$

Here $y \in \pm 1$ specifies the target class and $p \in [0, 1]$ is the estimated probability for the class with label $y = 1$, also known as objectness. The parameter γ is used to down-weight easy negatives, $\alpha_t \in [0, 1]$ is additional balancing factor and defined as α for class 1 and $1-\alpha$ for class -1.

It was shown in [18] that such formulation naturally emphasizes hard samples and prevents a vast number of easy negatives from dominating the loss. In this paper, we used $\gamma = 2$ and $\alpha = 0.25$, because this configuration demonstrated superior results in works [18, 37].

Adaptive feature pooling

At last, we further pushed the performance of the model by pooling and fusing ROI features from each FPN level. It allows each proposal to access both rich context information from deep pyramid levels and low-level features with high localization accuracy from the early levels. Here we did not follow CARoI pooling procedure [10], because concatenating already deep features and its further processing is too time-consuming. Instead, we have adopted adaptive feature pooling described in the reference [20] and directly fuse features from multiple levels via the max function. In comparison to the work [20], we did not include any additional layers before max fusion to keep inference performance near real time.

Our final detection model is schematically shown on the Fig. 2.

Tracking module

The tracking module of the proposed system answers two questions: Where did each vehicle come from, and where did it go? To answer these questions, we first track each detected vehicle across multiple frames and then classify its movement direction.

For vehicle tracking, we adopted SORT tracker [2] as it has a good speed-precision trade-off. We did not consider more recent works that fuse motion and appearance clues. Even providing a better performance, computing visual features for our crowded scenes is computationally expensive.

SORT provides the required functionality to track objects across multiple frames. On the next step, given the complete or incomplete trajectory of the car, we need to classify its general movement direction. To do so, one might train a shallow classifier

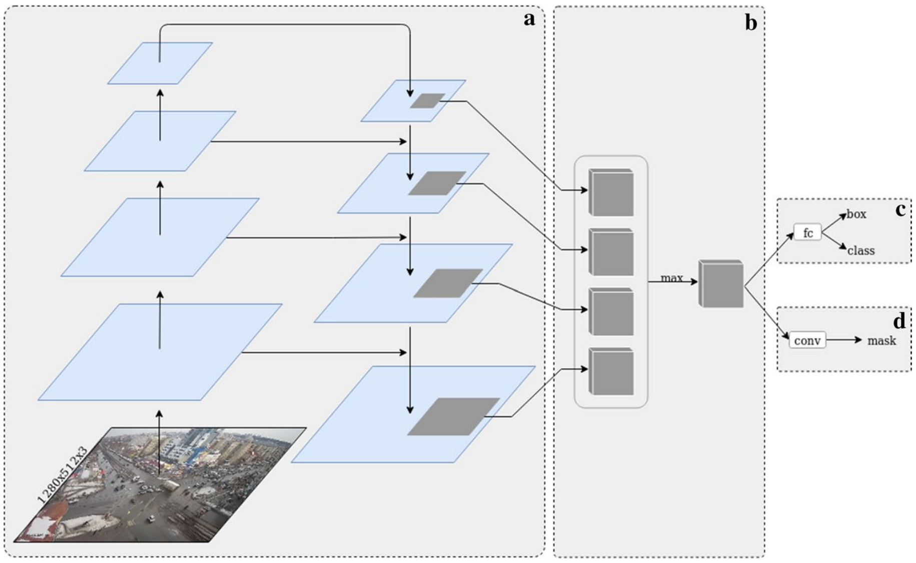

Fig. 2 Conceptual illustration of our detection network: **a** backbone with feature pyramid network [17]; **b** adaptive feature pooling [20]; **c** box branch; **d** mask branch. Note, that RPN and channel dimension of the feature maps are omitted for brevity

similar to the work [38]. However, we found that even a straightforward, heuristic-based approach works sufficiently well for us.

Our method includes several steps. First, we divide the crossroad into several non-overlapping regions. Each region defines either an adjacent road or the center of the crossing. Assuming that we already have tracking results for the whole video sequence, we approximate the vehicle position in each frame by the center of the bottom edge of the bounding box. Now, given a complete trajectory of each vehicle in the image coordinates, one can assign each path point to one of the crossroad sections (or void if nothing). It transforms the movement trajectories into a sequence of the visited regions.

Despite possible deviations, this sequence should generally be dividable into three major parts: source road, crossing, and target road. A couple of source and target roads uniquely describes the vehicle movement direction. We also need a way to deal with unfinished trajectories. It is a bit trickier, as they are usually the result of the detector fault or object disappearance due to the long-term occlusions. To address this problem, we merge all unfinished trajectories, i.e. one that ends or starts on the crossing center, within a short temporal and spatial window.

Thus, we first merge all incomplete trajectories and discard sequences that do not satisfy the assumption above. Then we split resulting sequences into two parts: before and after the crossroad. The most common source and target regions form a movement direction label. All trajectories that are either not logically justified or include void as a source or target region are discarded.

Experiments

The developed system was evaluated in two ways. First, we trained and tested the detection network alongside all proposed improvements on our new dataset. Second, we assessed the overall performance in the traffic flow estimation problem. To do this, we used the system to process surveillance data at peak hours. The results of the number of passing vehicles for the 4 busy driving directions were compared to the ground true statistics gathered by people.

All experiments was run on Ubuntu 18.04 with Nvidia RTX 2080 Ti GPU, 12 CPU cores (Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz) and 32 GB RAM. We built our solution on top the of the official Facebook Detectron realization [23].

Data processing details

We trained and validated the detection model solely on the collected dataset. Specifically, we used 786 images for training and 196 for validation, e.g. 80/20 split.

Training and validation images were preprocessed to limit the influence of the irrelevant regions. We masked each image in a way to leave only the roadway visible, i.e. parking areas that are adjacent to the road, uncategorized objects and distant sections of the road were cropped. Due to the limited GPU memory, we also rescaled each image to the 1280×512 resolution.

To improve robustness we augmented training frames by flipping, contrast and brightness adjustment, simulated occlusion and camera-specific artefacts. Occlusions were generated by either simple masking part of the sufficiently large vehicles (i.e. objects that occupy at least 40 pixels) or its extreme blurring. To reproduce camera faults, we blurred tall random horizontal sections near the image center.

Detection network details

We used ResNet-50 model [43] as a backbone together with the default FPN configuration from the Detection [23], as it provides a good trade-off between speed and detection quality.

Stochastic gradient descent with momentum was used to train the network. The initial learning rate was set to $1e-3$ and reduced by a factor of 10 after 20k and 27k iterations. In total, training was terminated after 30k iterations. We fixed a batch size equal to 4 for all experiments.

To speed up convergence, we used weights pre-trained on the MS COCO dataset [19]. Before training the whole network, we finetuned last layers of the RPN and classification/mask branches for 1k iterations with learning rate linearly decaying from $25e-3$ to $1e-3$.

We refer readers to the source code for more specific implementation details.

Evaluation protocol

To thoroughly validate performance of the detection model and impact of each proposed improvement, we report mean average precision (AP), mean average precision at IOU 0.5 ($AP_{0.5}$) and mean average precision at IOU 0.75 ($AP_{0.75}$) as they are used in several major detection benchmarks [5, 19, 39]. For completeness, besides AP for bounding boxes (AP^{bb}) we also report average precision for instance segmentation (AP^{segm}).

For the task of the traffic flow estimation, we report mean absolute error and mean absolute percentage error. We used statistics gathered by humans as a ground true data.

Table 2 Influence of the proposed architectural decisions on the detection network efficiency

	Baseline					Overall gain
Mask branch	✗	✓	✓	✓	✓	
Anchors optimization	✗	✗	✓	✓	✓	
Focal loss	✗	✗	✗	✓	✓	
Adaptive feature pooling	✗	✗	✗	✗	✓	
AP^{bb}	57.3	58.2	58.4	58.9	*59.2*	*+1.9*
AP^{bb}_{50}	79.0	79.7	79.7	79.7	*80.3*	*+1.3*
AP^{bb}_{75}	67.5	68.8	68.9	69.4	*70.0*	*+2.5*
AP^{segm}	–	56.1	56.4	56.7	*57.2*	*+1.1*
AP^{segm}_{50}	–	78.9	79.4	79.5	*80.2*	*+1.3*
AP^{segm}_{75}	–	68.0	*69.2*	68.8	69.1	*+1.1*

Italic values indicate the performance gain relative to the Faster R-CNN baseline when applying all the proposed improvements

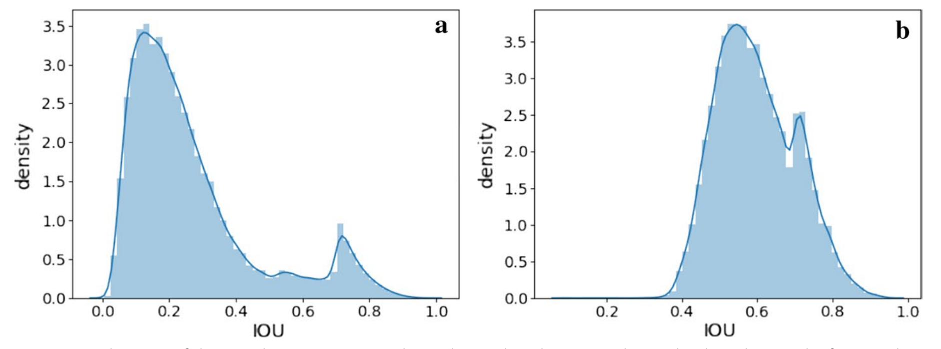

Fig. 3 Distribution of the IOU between ground true bounding boxes and matched anchors: **a** before anchors optimization, **b** after optimization

Results and discussions

Detection performance

The Table 2 shows the results of the baseline Faster R-CNN model alongside the influence of each proposed improvement. It is clear that all used techniques provide a solid advance relative to the baseline. However, obtained results are worse by approximately 10 to 20 AP points than results of the similar top-ranked models on a popular UA-DETRAC and KITTI datasets [6, 39]. It supports the conclusions that our target data is more challenging than well-studied counterparts.

We expected more gain from the anchors optimization as Fig. 3 clearly states that anchors chosen by grid search procedure provide better coverage of the ground true boxes. The reasons might lie in the fact that we used pre-trained weights, i.e. our network is initially biased towards objects whose shape is different from the target one.

Our network runs at 15 frames per second and can process video sequences in real-time by operating on each second frame. Figure 4 shows some visual results of the model on the validation dataset. One might further improve the quality by replacing ResNet-50 backbone with massive ResNeXt-101 model [41] or by constructing a heavier feature

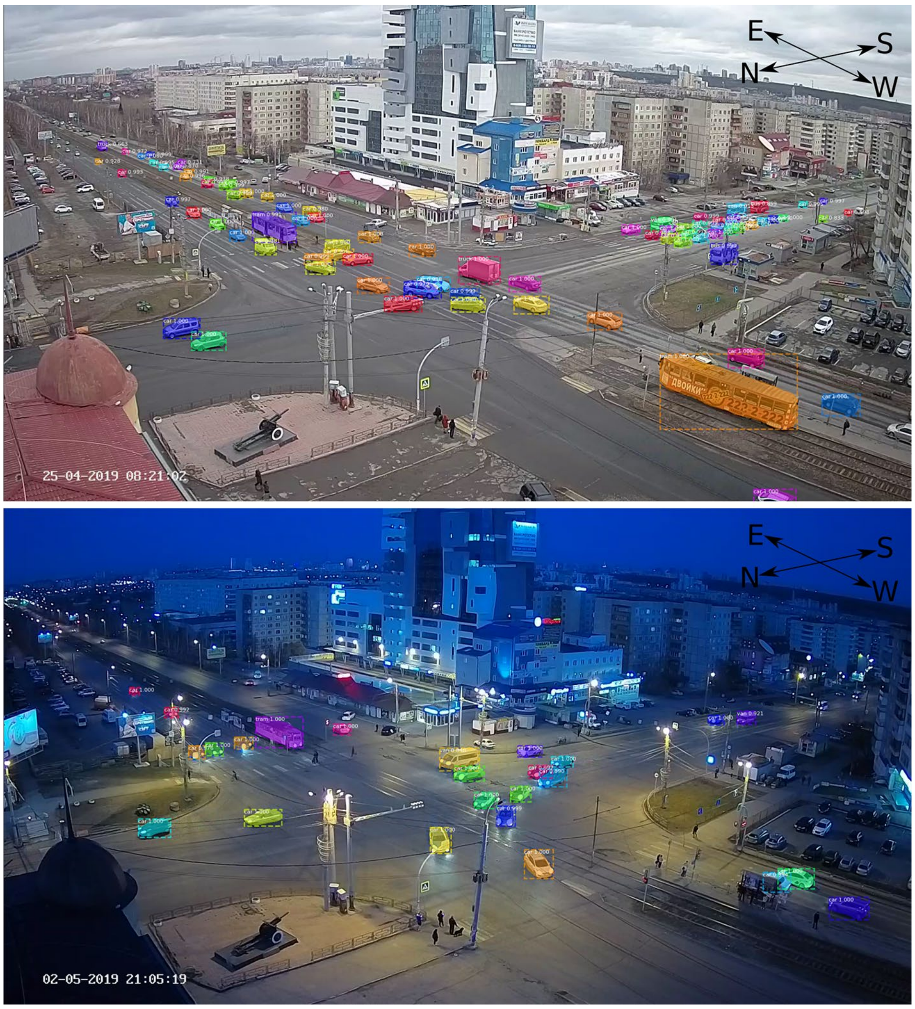

Fig. 4 Results of the proposed detection network. Driving directions are shown in the upper right corner. Better viewed with zoom

pyramid network. However, it would make the network completely inappropriate for near real-time video processing.

System performance

Evaluation results for the traffic flow estimation task are presented in Fig. 5. Table 3 shows mean absolute and mean absolute percentage error compared to the human results. Given the overall performance, we can conclude that the developed system, although showing promising results, is still behind the human level. However, our solution already has applied value. For instance, it can and will be used to optimize traffic light timings or detect anomalies in crossroad throughput.

Additional investigation of the typical errors suggests that most of them are the result of strong and long-lasting occlusions between vehicles in a dense traffic stream. For

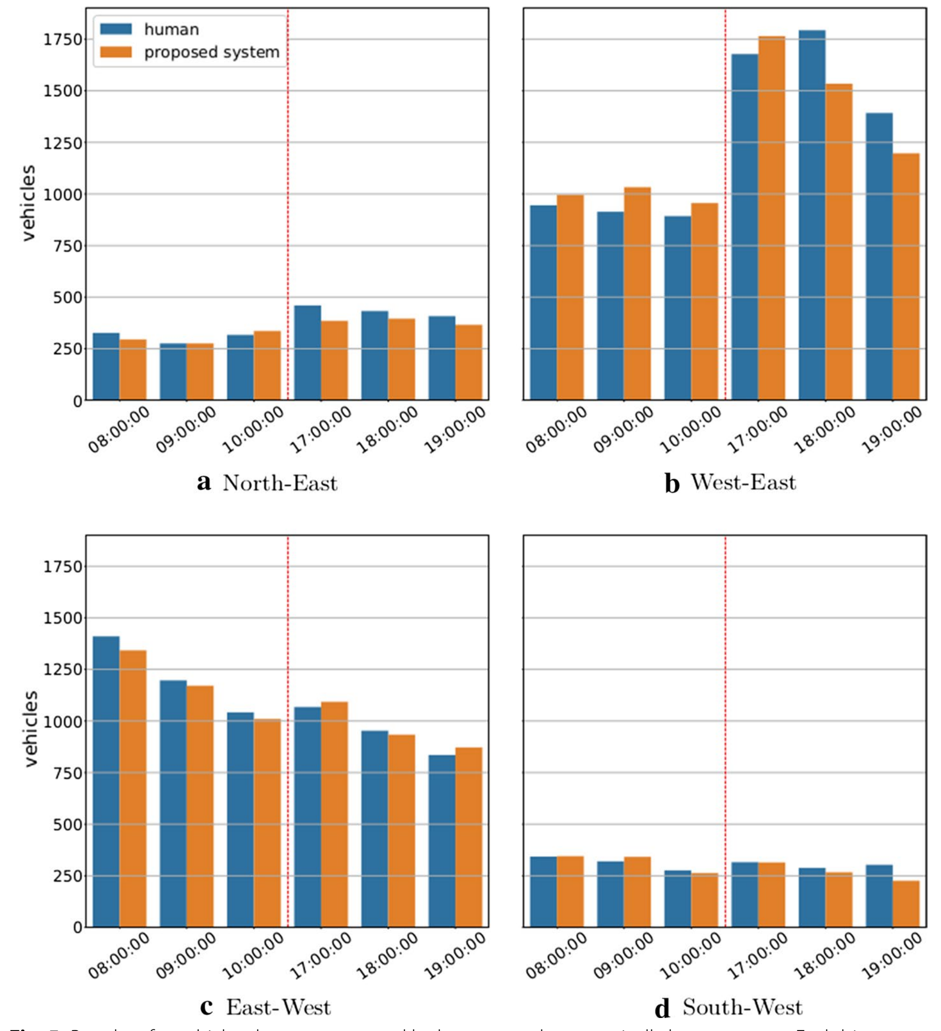

Fig. 5 Bar plots for vehicles that were counted by humans and automatically by our system. Each bin includes statistic for a single hour, four separate driving directions are reported

Table 3 Mean absolute error and mean absolute percentage error per hour for the traffic flow estimation task

Direction	Mean absolute error	Mean absolute percentage error
North–East	34.17	8.49
West–East	129.0	9.85
East–West	34.5	3.14
South–West	22.67	7.53
Average	55.08	7.25

Metrics are detailed for four movement directions

example, during the movement, trams partially overlay from one to two lanes for the west-east movement direction. Many cars overlap while turning, waiting in the center of the intersection for a free window. This problem can be addressed by improving the tracking module with special techniques for instances re-identification based on appearance clues. However, as mentioned earlier, existing approaches have a high computational burden and are not applicable to real systems. Designing efficient algorithms for vehicle re-identification remains an open question.

Conclusions

In this work, we focused on the problem of traffic flow estimation with data from a video surveillance camera. As a case study, we considered one of the busiest crossroads in Chelyabinsk, Russia. Being practically important, this subject area is still in early development. Up to date, there are just a few works that aim to automatically process surveillance video data for traffic analysis.

To address the posed problem, we used SORT tracker and enhanced state-of-the-art Faster R-CNN detector. The proposed system can count and classify vehicles by driving directions with a mean percentage error that is less than 10%. Achieved performance indicates that there is a room for improvement, yet it is already enough to solve some real-world problems. Our solution will be used further by city authorities to optimize crossroad throughput. The horizontal scaling would make possible even more fruitful optimization of the traffic flows throughout the city.

To the best of our knowledge, there are no publicly available datasets or benchmarks that cover conditions similar to ours. To train and evaluate the detector, we gathered 982 video frames with more than 60,000 labelled vehicles in total. Collected data covers only a single crossroad and is not applicable as a general purpose benchmark. However, other studies might use it as an additional training data or challenging test set. Our data is diverse in terms of the presented weather conditions, traffic situation, vehicle shapes, scales, and classes.

Despite achieved results, there are still many problems to face. Here we did not cover problems of vehicle re-identification and fine-grained classification, possible optimizations of convolutional neural networks for efficient and cheap inference. Separately, there are tasks related to the efficient surveillance data movement and storage. We consider these problems as possible directions for future researches.

Abbreviations
CCTV: closed-circuit television; MOT: multi-object tracking; CNN: convolutional neural networks; ROI: regions of interest; CARoI: context-aware RoI; DA: data association; MHT: Multiple Hypothesis Tracking; JPDAF: Joint Probalistic Data Association Filter; FPN: feature pyramid network; IOU: intersection over union; RPN: region proposal network; FL: focal loss; AP: mean average precision.

Acknowledgements
Not applicable.

Authors' contributions
SI supervised data collection and processed the dataset. AF and KN implemented the proposed system, performed the experiments and analyzed test results. VS and AM formulated the problem, suggested initial ideas and influenced the shape of the research. AF wrote an initial draft of the manuscript. All authors suggested related works, discussed the structure of the paper and results. All authors read and approved the final manuscript.

Funding
The work was supported by Act 211 Government of the Russian Federation, contract No. 02.A03.21.0011.

Availability of data and materials
All code is available on-line, https://github.com/alnfedorov/traffic-analysis. Data access will be provided upon a request.

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

Author details
[1] South Ural State University, 454091 Chelyabinsk, Russia. [2] The Institute of State and Law of The Russian Academy of Sciences, Moscow 119019, Russia.

Received: 18 May 2019 Accepted: 23 July 2019
Published online: 07 August 2019

References

1. Anisimov D, Khanova T. Towards lightweight convolutional neural networks for object detection. In: 2017 14th IEEE international conference on advanced video and signal based surveillance (AVSS). 2017, p. 1–8. https://doi.org/10.1109/AVSS.2017.8078500, http://ieeexplore.ieee.org/document/8078500/, arXiv:1707.01395.
2. Bewley A, Ge Z, Ott L, Ramos F, Upcroft B. Simple online and realtime tracking. In: 2016 IEEE international conference on image processing (ICIP). vol. 2016-Aug, p. 3464–8. IEEE. 2016. https://doi.org/10.1109/ICIP.2016.7533003, http://ieeexplore.ieee.org/document/7533003/.
3. Brooks J. COCO annotator. 2019, https://github.com/jsbroks/coco-annotator/.
4. Chu P, Ling H. FAMNet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. 2019. arXiv:1811.07258arXiv:1904.04989.
5. Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B. The cityscapes dataset for semantic urban scene understanding. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR). 2016, p. 3213–23. IEEE. https://doi.org/10.1109/CVPR.2016.350, arXiv:1604.01685, http://ieeexplore.ieee.org/document/7780719/.
6. Geiger A, Lenz P, Stiller C, Urtasun R. Vision meets robotics: the KITTI dataset. Int J Rob Res. 2013;32(11):1231–7. https://doi.org/10.1177/0278364913491297.
7. Girshick R. Fast R-CNN. In: Proceedings of the IEEE international conference on computer vision 2015 Inter, 2015, p. 1440–8. https://doi.org/10.1109/ICCV.2015.169.
8. Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accu- rate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2012, p. 580–7, http://arxiv.org/pdf/1311.2524v3.pdf.
9. He K, Gkioxari G, Dollar P, Girshick R. Mask R-CNN. In: 2017 IEEE international conference on computer vision (ICCV). vol. 2017 Oct, 2017, p. 2980–8. IEEE. https://doi.org/10.1109/ICCV.2017.322, http://ieeexplore.ieee.org/document/8237584/.
10. Hu X, Xu X, Xiao Y, Chen H, He S, Qin J, Heng PA. SINet: A scale-insensitive convolutional neural network for fast vehicle detection. In: IEEE transactions on intelligent transportation systems. vol. 20, no. 3. 2019, p. 1010–9. https://doi.org/10.1109/TITS.2018.2838132, https://ieeexplore.ieee.org/document/8478157/.
11. Huang J, Rathod V, Sun C, Zhu M, Korattikara A, Fathi A, Fischer I, Wojna Z, Song Y, Guadarrama S, Murphy K. Speed/accuracy trade-offs for modern convolutional object detectors. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR). 2017, p. 3296–7. IEEE. https://doi.org/10.1109/CVPR.2017.351, arXiv:1611.10012http://ieeexplore.ieee.org/document/8099834/.
12. Jung H, Choi MK, Jung J, Lee JH, Kwon S, Jung WY. ResNet-based vehicle classification and localization in traffic surveillance systems. In: 2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW). 2017, p. 934–40. IEEE. https://doi.org/10.1109/CVPRW.2017.129, http://ieeexplore.ieee.org/document/8014863/.
13. Kim C, Li F, Ciptadi A, Rehg JM. Multiple hypothesis tracking revisited. In: 2015 IEEE international conference on computer vision (ICCV). vol. 22. 2015, p. 4696–704. IEEE. https://doi.org/10.1109/ICCV.2015.533, http://ieeexplore.ieee.org/document/7410890/.
14. Li C, Dobler G, Feng X, Wang Y. TrackNet: simultaneous object detection and tracking and its application in traffic video analysis. 2019, p. 1–10, arXiv:1902.01466.
15. Li S, Lin J, Li G, Bai T, Wang H, Pang Y. Vehicle type detection based on deep learning in traffic scene. Procedia Comput Sci. 2018;131:564–72. https://doi.org/10.1016/j.procs.2018.04.281.
16. Li S. 3D-DETNet: a single stage video-based vehicle detector. 2018, arXiv:1801.01769.
17. Lin TY, Dollar P, Girshick R, He K, Hariharan B, Belongie S. Feature pyramid networks for object detection. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR). vol. 2017 Jan, 2017, p. 936–44. IEEE. https://doi.org/10.1109/CVPR.2017.106, http://ieeexplore.ieee.org/document/8099589/.
18. Lin TY, Goyal P, Girshick R, He K, Dollar P. Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. 2017 Oct, p. 2999–3007. https://doi.org/10.1109/ICCV.2017.324.
19. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. Microsoft COCO: common objects in context. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). vol. 8693 LNCS, 2014, p. 740–55. https://doi.org/10.1007/978-3-319-10602-1_48.
20. Liu S, Qi L, Qin H, Shi J, Jia J. Path aggregation network for instance segmentation. In: 2018 IEEE/CVF conference on computer vision and pattern recognition. 2018, p. 8759–68. IEEE. https://doi.org/10.1109/CVPR.2018.00913, https://ieeexplore.ieee.org/document/8579011/.
21. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC. SSD: single shot multibox detector. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics) 9905 LNCS. 2016, p. 21–37. https://doi.org/10.1007/978-3-319-46448-0_2.
22. Luo W, Yang B, Urtasun R. Fast and furious: real time end-to-end 3D detection, tracking and motion forecasting with a single convolutional net. In: 2018 IEEE/CVF conference on computer vision and pattern recognition. 2018, p. 3569–77. IEEE. https://doi.org/10.1109/CVPR.2018.00376, https://www.scopus.com/inward/record.uri?eid=2-s2.0-84956766303&partnerID=40&md5=123d54008fbf65570432b00879599f56https://ieeexplore.ieee.org/document/8578474/.

23. Massa F, Girshick R. maskrcnn-benchmark: Fast, modular reference implementation of instance segmentation and object detection algorithms in PyTorch; 2018. https://github.com/facebookresearch/maskrcnn-benchmark.
24. Peppa MV, Bell D, Komar T, Xiao W. Urban traffic flow analysis based on deep learning car detection from cctv image series. Int Arch Photogramm Remote Sens Spat Inf Sci. 2018;42(4):565–72. https://doi.org/10.5194/isprs-archives-XLII-4-499-2018.
25. Rathore MM, Son H, Ahmad A, Paul A. Real-time video processing for traffic control in smart city using Hadoop ecosystem with GPUs. Soft Comput. 2018;22(5):1533–44. https://doi.org/10.1007/s00500-017-2942-7.
26. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR). 2016, p. 779–88. IEEE. https://doi.org/10.1109/CVPR.2016.91, arXiv:1506.02640http://ieeexplore.ieee.org/document/7780460/.
27. Redmon J, Farhadi A. YOLO9000: Better, faster, stronger. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR). 2017, p. 6517–25. IEEE. https://doi.org/10.1109/CVPR.2017.690, arXiv:1612.08242http://ieeexplore.ieee.org/document/8100173/.
28. Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans Pattern Anal Mach Intell. 2017;39(6):1137–49. https://doi.org/10.1109/TPAMI.2016.2577031.
29. Rezatofighi SH, Milan A, Zhang Z, Shi Q, Dick A, Reid I. Joint probabilistic data association revisited. In: 2015 IEEE international conference on computer vision (ICCV). 2015, p. 3047–55. No. December, IEEE. https://doi.org/10.1109/ICCV.2015.349, http://ieeexplore.ieee.org/document/7410706/.
30. Sang J, Wu Z, Guo P, Hu H, Xiang H, Zhang Q, Cai B. An improved YOLOv2 for vehicle detection. Sensors. 2018;18(12):4272. https://doi.org/10.3390/s18124272.
31. Sochor J, Špaňhel J, Herout A. BoxCars: improving vehicle fine-grained recognition using 3D bounding boxes in traffic surveillance. 2017, p. 1–12. arXiv:1703.00686.
32. Sommer L, Acatay O, Schumann A, Beyerer J. Ensemble of two-stage regression based detectors for accurate vehicle detection in traffic surveillance data. 2019, p. 1–6. https://doi.org/10.1109/avss.2018.8639149.
33. Spanhel J, Sochor J, Makarov A. Vehicle fine-grained recognition based on convolutional neural networks for real-world applications. In: 2018 14th symposium on neural networks and applications (NEUREL). 2018, p. 1–5. IEEE. https://doi.org/10.1109/NEUREL.2018.8587012.
34. Sun X, Ding J, Dalla Chiara G, Cheah L, Cheung NM. A generic framework for monitoring local freight traffic movements using computer vision-based techniques. In: 5th IEEE international conference on models and technologies for intelligent transportation systems, MT-ITS 2017—proceedings. 2017; p. 63–8. https://doi.org/10.1109/MTITS.2017.8005592.
35. Tang Z, Wang G, Xiao H, Zheng A, Hwang JN. Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features. In: 2018 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW). vol. 2018 June, 2018, p. 1080–7. IEEE. https://doi.org/10.1109/CVPRW.2018.00022, https://ieeexplore.ieee.org/document/8575384/.
36. 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. IEEE. https://doi.org/10.1109/ICME.2017.8019461.
37. Wang X, Cheng P, Liu X, Uzochukwu B. Focal loss dense detector for vehicle surveillance. In: 2018 international conference on intelligent systems and computer vision (ISCV). vol. 2018 May, 2018, p. 1–5. IEEE. https://doi.org/10.1109/ISACV.2018.8354064.
38. Wei Y, Song N, Ke L, Chang Mc, Lyu S. Street object detection/tracking for AI city traffic analysis. In: 2017 IEEE smartworld, ubiquitous intelligence & computing, advanced & trusted computed, scalable computing & communications, cloud & big data computing, internet of people and smart city innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). 2017, p. 1–5. No. i. IEEE. https://doi.org/10.1109/UIC-ATC.2017.8397669.
39. Wen L, Du D, Cai Z, Lei Z, Chang MC, Qi H, Lim J, Yang MH, Lyu S. UA-DETRAC: a new benchmark and protocol for multi-object detection and tracking. 2015. arXiv:1511.04136.
40. Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE international conference on image processing (ICIP). vol. 2017 Sept, 2017, p. 3645–9. IEEE. https://doi.org/10.1109/ICIP.2017.8296962.
41. Xie S, Girshick R, Dollár P, Tu Z, He K. Aggregated residual transformations for deep neural networks. In: Proceedings—30th IEEE conference on computer vision and pattern recognition, CVPR 2017. 2017-Jan, 2016, p. 5987–95. https://doi.org/10.1109/CVPR.2017.634, arXiv:1611.05431.
42. Zhang S, Wu G, Costeira JP, Moura JM. FCN-rLSTM: Deep spatio-temporal neural networks for vehicle counting in city cameras. In: Proceedings of the IEEE international conference on computer vision 2017-Oct, 2017, p. 3687–96. https://doi.org/10.1109/ICCV.2017.396.
43. Zhang S, Wu G, Costeira JP, Moura JM. Table of contents. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR). vol. 2017-Jan, 2017, p. v–liii. IEEE. https://doi.org/10.1109/CVPR.2017.4, http://ieeexplore.ieee.org/document/8099487/.
44. Zhu F, Lu Y, Ying N, Giakos G. Fast vehicle detection based on evolving convolutional neural network. In: 2017 IEEE international conference on imaging systems and techniques (IST). vol. 2018-Jan, 2017, p. 1–4. IEEE. https://doi.org/10.1109/IST.2017.8261505.