Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 149 (2019) 151–158

Procedia Computer Science

www.elsevier.com/locate/procedia

ICTE in Transportation and Logistics 2018 (ICTE 2018)

Automatic data labeling by neural networks for the counting of objects in videos

Ivars Namatevs[a,c,*], Kaspars Sudars[b], Inese Polaka[a]

[a]*Riga Technical University, Kalku str.1, Riga, LV-1050, Latvia*
[b]*Institute of Electronics and Computer Science, Dzerbenes str.14, Riga, LV-1006, Latvia*
[c]*Turiba University, Graudu str.68, Riga, LV-1058, Latvia*

Abstract

The paper proposes an efficient method for training a neural network to count moving objects in a video, while another neural network concurrently prepares a labeled dataset for the first one. The detection, tracking, and counting of objects is crucial for effective Intelligence Transportation Systems (ITS), which should reduce congestion and recognize traffic offenders on highways and in urban areas. Creation of labeled data for training a neural network is one of the essential prerequisites for successful application of supervised machine learning. In this paper, the experimental results of the automatic labeling and counting of vehicles under real world conditions are shown. The method shows that by using the Convolutional Neural Network (CNN), the computing power and speed-up time for training a Recurrent Neural Network (RNN) with a Long Short-Term Memory (LSTM) cell for counting moving objects can be decreased.

Keywords: Object detection; Data labeling; Convolutional neural network; Long short-term memory

1. Introduction

It is estimated that there will be 2 billion motor vehicles on the roads by 2030 [1]. Such a large number of vehicles raises concerns about how to ensure security and safety on highways and streets. Therefore, smart control systems and comprehensive traffic management will be the solution to deal with this large increase of vehicles.

* Corresponding author. Tel.: +371 26133567.
E-mail address: ivars@turiba.lv

1877-0509
10.1016/j.procs.2019.01.118

Security and safety on roads or streets will be dependent on a computer vision that includes smart observation systems. One of the main challenges for such systems will be to recognize vehicles in significant motion [2]. To solve this and other problems in detecting, and tracking objects, is what the Intelligent Transportation Systems (ITS) offers. Object detection is an area in computer vision that is emerging very fast. New algorithms, models and a deep learning approach outperform previous methods. Recently, there has been an accumulation of pre-trained models for object detection, e.g. YOLO [3]. Thus, there is no real problem in detecting an object in an image or a video. Today, the problem is, how to carry this out in a more effective and efficient way. The challenge is how to construct algorithms and models, which consume less computing time and memory, energy, as well as financial resources. It is not a secret that companies which have been involved in object detection have been looking, on the one hand, for inexpensive solutions, which can be carried out on standard computers and cameras, but, on the other hand, which should provide a high level of confidence.

2. Application domains of vehicle detection

Advancements in computer vision, deep neural networks and video technologies, together with the increase of vehicles in urban areas and on roads, provide interest in the ITS. These systems are designed to efficiently organize traffic, including traffic control centers, tracking and detection. The last one, object detection, is a salient approach in many domains, such as surveillance, robotics, human defense, retrieval, logistics, etc. [4]. This process involves the object detection ranging from vehicle detection to tracking people or animals in specific areas. Most recently, another crucial application for research is the detection of external environment objects by self-driving cars or flying drones' cameras. In particular, this should be done for the safety of pedestrians on the streets, in public shows and sport events. Due to the increasing availability of images from video and sensors, providing further enhancement of data proceedings and analytics, and the establishment of powerful deep learning algorithms, today`s object detection can be fast and effective. Despite the introduction of a new technologies and models in object detection, there are still many factors that can have an influence and that must to be considered when creating intelligent an object detection system, such as object occlusions, background clutter, camera placement and illumination changes [5]. This is not a complete list; there are other influential factors which can have a substantial impression on object detection, e.g. the rotation invariance, inter-class and intra-class variation, and multi-pose object detection [6]. Based on the above mentioned, object detection is a core problem in computer vision for the vehicle detection in the ITS. Moreover, vehicle detection is an important branch of image processing studies.

The exploration of vehicle detection is mainly divided into motion-estimation-based techniques and appearance-based techniques [5]. Motion-based-techniques includes: background subtraction method [7, 8], interframe difference method [9], Gaussian scale mixture method [10], optical flow estimation method [11], median filter method [12], and Kalman filter method [13]. Background subtraction is the most popular approach for vehicle detection tasks using a static camera. In this method, the background is subtracted from a current frame of the video to get foreground blobs that corresponds to the moving vehicles [2]. Generally, in the background-based method the object of detection is distinguished from the background by certain parameters, e.g. color or intensity of the pixels. The focus of appearance-based techniques is based on the detection algorithm which detects and classifies the vehicle target in the actual traffic video or picture. Its main difficulty lies in the video frame or picture of the vehicle target, which will change due to lighting, angle of view, the interior of the vehicle and other changes, as well as the characteristics of different types of vehicles. Appearance-based techniques include: the feature-based method [14], the part-based model method [15] the Scale-Invariant Feature Transform (SVIF) [16], the Speed Up Robust Feature (SURF) [17], the Gradient Location and the Orientation Histogram (GLOH) [18]. Alongside with these techniques, several virtual detections line-based (VDL) methods have been proposed for counting vehicles that pass a so called a virtual line of the frame of a video [2].

The breakthrough and rapid adoption of deep learning in 2012, viz. powerful Graphical Processing Units (GPU) and availability of large datasets, brought into existence state-of-the-art and high accurate object detection algorithms and methods, mainly accomplished by the convolutional neural network (CNN) based approaches. For instance, R-CNN [19], Fast-RCNN [20], Faster-RCNN [21], Mask R-CNN [22] and fast yet highly accurate ones like SSD [23] and YOLO [3, 24, 25]. As was mentioned before, CNNs are the main building block for most of the

object detection tasks. In deep CNN, object detection is reframed as a single regression problem from image pixels to bounding box coordinates. Vehicle detection has been associated as the problem of sequence modeling.

Recurrent Neural Network (RNN), especially with Long Short-Term Memory (LSTM) cell, has recently been applied in the sequence modeling tasks, such as video capturing [26] and speech recognition [27], and human motion prediction [28]. Based on the assessment of existing approaches, the goal of vehicle detection and their counting is to develop a compliant method for detection of moving object where data labeling for training should be realized automatically. Therefore, we could use our understanding of theses varieties to connect the deep neural network and object detection technique together with the right solution for automatic data labeling for the counting moving objects.

3. Deep convolutional neural networks

To create an intelligent object counting system, first we need to detect and classify the moving object. Object detection refers to the capability of computers and software systems to locate objects in an image and identify each object, e.g. vehicles, pedestrians, etc. Deep CNNs have been widely used for image classification, object classification, localization, and object detection [29]. Layered architecture of CNN, consisting of pooling layers, feature maps, and fully connecting layers, can describe the actions and relationships to objects according to image context [7, 30]. The object detection and localization tasks can be trained by using ImageNet dataset [31], but the MS COCO dataset for the image captioning tasks [32].

There are several traditional detection and localization systems. Systems like deformable parts models (DPM) [20] use a sliding window approach where the classifier is run at evenly spaced locations over the entire image. Although, this approach is recognized as a standard, there are two main caveats. First, it requires a lot of computing power which leads to expensive computing. Second, the bounding boxes do not match high accuracy, especially when the object is not rectangular. A faster and more accurate convolution has been realized by using the idea to divide the image into multiple grids. This approach detects the objects straight from image pixels to bounding box coordinates and class probabilities. The network uses features from the entire image to predict each bounding box [24, 25]. Briefly, the algorithm is as follows. The input image has been split into k x k number of grids. The output vector will be of length k x k x (c + 5), where c is the number of unique objects in our data. If the centroid of an object falls into a grid, that grid cell is responsible for detecting that object. This is an important feature so as not to allow the counting of one object multiple times in different grids. Each bounding box consists of 5 predictions: *x, y, w, h,* and confidence. The x, y coordinates represents the center of the box relative to the bounds of the grid cell. The width and height are predicted relative to the whole image. Finally, the confidence prediction represents the intersection over union (IOU) between the predicted box and any ground truth box. The confidence is defined as Pr(Object) * IOU_{pred}^{truth} [3, 24]. These confidence scores reflect how confident the model is that the box contains an object and also how accurate it thinks its predicted the box is. If no object exists in that cell, the confidence score should be zero. Each grid cell also predicts C conditional class probabilities, Pr(Class$_i$|Object) [3, 24].

More recent approaches can be undertaken, such as a detection method with a sequential Monte Carlo (SMC) filter to automatically select samples associated with the right label by using a faster R-CNN detector [21] or a co-occurrence matrix to extract features can be undertaken [33].

4. Recurrent neural network with long short-term memory

When the objects have been detected and classified, the next step is vehicle counting, to create an adaptable automatic data labeling approach. For this purpose, the RNN-LSTM neural network has been used.

The RNN processes information from the current input vector $x^{(t)}$ by incorporating it into the state $h^{(t)}$ that is passed forward through time to current output $\hat{y}^{(t)}$.

$$h^{(t)} = g_h\left(W_{ih}x^{(t)} + W_{hh}h^{(t-1)} + b_h\right). \quad (1)$$

As we are not using the deep RNN, the stack of recurrently connected layers N is 1 and the architecture reduces to a simple next step prediction.

$$h_1^{(t)} = g_h\left(W_{ih_1}x^{(t)} + W_{h_1h_1}h_1^{(t-1)} + b_{h_1}\right). \quad (2)$$

Where the W terms denote weight metrices, the b terms denote bias vectors, and g_h is a hidden layer vector or activation function. The output space is computed as follows:

$$\hat{y}^{(t)} = b_y + \sum_n^N W_{hy}h^{(t)}. \quad (3)$$

Therefore, RNN defines a function, parametrized by the weight matrices, from input histories $x_{1:t}$ to output vectors $\hat{y}^{(t)}$ [34].

The gradient of error between the predicted output $\hat{y}$ and the correct result y can be backpropagated through time as described in [35]. Nevertheless, the vanishing and exploding problem for RNNs has existed. This was independently discovered by separate researchers. One of often used solutions in practical applications is gated RNNs that have derivatives that neither vanish nor explode [36]. This means that the network accumulates information over a long duration. Once that information has been used, the network forgets the old state. It has been found that the LSTM architecture, which uses purpose based purpose-built memory cells to store information, is better at finding and exploiting long range dependencies in the data [37]. The LSTM math composite function are:

$$i^{(t)} = \sigma\left(W_{xi}x^{(t)} + W_{hi}h^{(t-1)} + W_{ci}c^{(t-1)} + b_i\right). \quad (4)$$

$$o^{(t)} = \sigma\left(W_{xo}x^{(t)} + W_{ho}h^{(t-1)} + W_{co}c^{(t)} + b_o\right). \quad (5)$$

$$f^{(t)} = \sigma\left(W_{xf}x^{(t)} + W_{hf}h^{(t-1)} + W_{cf}c^{(t-1)} + b_f\right). \quad (6)$$

$$c^{(t)} = f^{(t)}c^{(t-1)} + i^{(t)}tan\,h\left(W_{xc}x^{(t)} + W_{hc}h^{(t-1)} + b_c\right). \quad (7)$$

$$h^{(t)} = o^{(t)}tan\,h\left(c^{(t)}\right). \quad (8)$$

Where *i, o, f,* and *c* are respectively the input gate, output gate, forget gate, internal cell input activation vectors, all of which are the same size as the hidden vector, σ is the logistic sigmoid function. The subscript *t* indicates the *t*-th step in LSTM. $x^{(t)}$ denotes the input data. $h^{(t)}$ stands for hidden state. The input gate, output gate, and forget gate receive the current input vectors $x^{(t)}$ and $h^{(t-1)}$ as inputs. The input node sums all its inputs and uses nonlinear function for an activation value. This value is multiplied by the activation from the input gate $i^{(t)}$, which may be 0 or 1. The input gate decides if the input nodes output is propagated to internal cell $c^{(t)}$. This cell has a recurrent edge to itself with weight value 1, which allows the unimpeded propagation of error through time. The forget gate $f^{(t)}$ triggers the deletion of the internal state $c^{(t)}$. The output gate $o^{(t)}$ decides if the output of LSTM is propagated further [31]. The LSTM) can be mathematically defined by equations (4-8).

5. Data labeling modes

Data labeling is a vital stage of data preprocessing in supervised learning. Each mistake or inaccuracy in this process can negatively affect a dataset`s quality. Moreover, the overall performance of a predictive model can be destroyed and lead to misinterpretation. Having this in mind, the ground truth in the object detection problem is that we need such an algorithm that could label the discrete data. There are three options to label the data: manual labeling, semi-automatic labeling, and automatic labeling. To choose the right option one must take into consideration the type of data. Data like vehicle image, voice and text are usually labeled by labor pools. This means there are two choices to do this: first, outsourcing it or, second, annotating label images internally by yourself, i.e. set up a data labeling team. Today there is a trend to arrange and organize the data in a more cutting-edge labeling

process, i.e to set upon on some features that can make the data labeling semi-automatic or to delegate this task automatically to machines. R. Kadikis [38] declared three modes of data labeling for vehicle detection in computer vision. The first mode, the output label corresponds to the number of objects on the VDL. The second labeling mode codes how many objects had left the line at the current frame. Finally, the third labeling mode is a combination of the two previous ones. For advantages and disadvantages of all data labeling modes see Table 1.

Table 1. Comparison of data labeling modes.

Data Labeling Mode	Advantage	Disadvantage
1.The output label corresponds to the number of moving objects on the VDL in a frame.	Easy recognition of the first entering moving object in a frame.	Two or more closely following objects might create a sequence of label (1,1, …, 1). The objects are overlapping.
2. The output label corresponds to how many objects had left the VDL in a frame.	Allowing detecting and counting the unambiguous object that fully crossed the line. Easy to manually label the training data since labeler only needs to add one per object, not per every frame.	The labeled dataset is unbalanced since in many practical applications most of the labels are zero.
3.The output label corresponds to how many moving objects have left VDL in the last *k/2* frames.	The human labeler only marks the frames where the objects have left the detection line.	The human labeling is tedious and time consuming.

Obviously, to create a valuable and precise vehicle counting system, a fine-grained vehicle classification of the image dataset is necessary [39] . Next, the current process of data preparation is not only painstaking but is susceptible to errors as it is done manually. Finally, labelling outstanding accuracy, setup time and speed are those factors that must be considered for training data sets. Having taken the above into account, we propose an approach that belongs to the automation of the data labeling process using one neural network which must diagnose by itself the right data labeling for training another neural network.

6. Automatic data labeling for vehicle detection

Our network architecture is inspired by the YOLO model for image detection and the RNN-LSTM. We use the YOLO neural network for image classification and recognition of the bounding boxes in video. The YOLO network consists of 24 convolutional layers followed by 2 fully connected layers. In our experiment we have used the video data from a surveillance camera which acquires approximately 8 FPS, resolution 640 x 480. For automatic labeling, the video stream was preprocessed, downloaded and stored in a 2 minute long video. Next, the position and length of VDL in frames of the video must be defined. The VDL is a line whose position is usually perpendicular to the motion of the vehicles and is independent of the frames. The aim of such a line is to detect the moving objects that cross it. Fig.2 depicts the virtual detection line in the example frames. The unlabeled videos of vehicles which arrived in or drove out from the parking lot were acquired from open streaming security cameras. Fig.1 shows example frames.

Fig. 1. (a) Video frame without vehicle; (b) Video frame with one vehicle on VDL; (c) Video frame with two vehicles on VDL.

The detector line is *n* pixels long and one-pixel high line. To improve the efficiency only one-line pixels are processed, while the rest of the frame is not. This is the main advantage of our proposed approach, which definitely gives economy of time and resources during the automatic data labeling process. In the practical application, it is crucial for the moving objects to be correctly counted (output of the RNN-LSTM) to determine the localization of the bounding box in the grid cell. For this purpose, we used the YOLO network. Fig.2 shows the pattern of the bounding box in a 2D coordinate system for automatic data labeling.

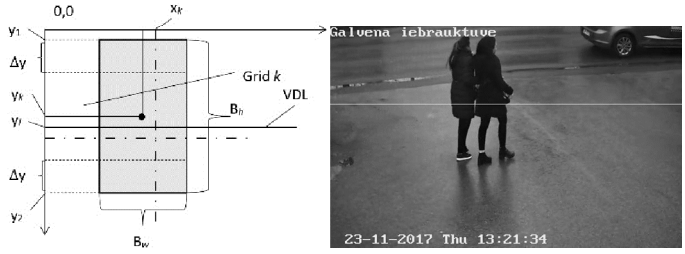

Fig. 2. (a) Scheme of the bounding box for automatic data labeling; (b) Video frame with two people on VDL.

For instance, we have a bounding box B_w x B_h provided by YOLO whose centroid is in *k* grid. VDL is as a pixel line determined and located as y_l. The lower and upper and positions of the bounding box on *y* axes are denoted y_1 and y_2 respectively. With Δy we have defined the region of not detection of the bounding box. Therefore, the automatic data labeling (vehicle) can be considered according to equation (8).

$$Data\ labeling = y_2 - \Delta y < y_l < y_1 + \Delta y, \tag{9}$$

where

$$\Delta y = 0{,}1B_h \tag{10}$$

The total time of the automatic data labeling with YOLO was 318 minutes, during which 366 vehicles were counted. The reason for providing conditions according to equation (10) is to decrease the error rate by 20%. To realize vehicle the vehicle counting function data after the automatic data labeling processes are forwarded to the input of the RNN-LSTM neural network, see Fig. 3.

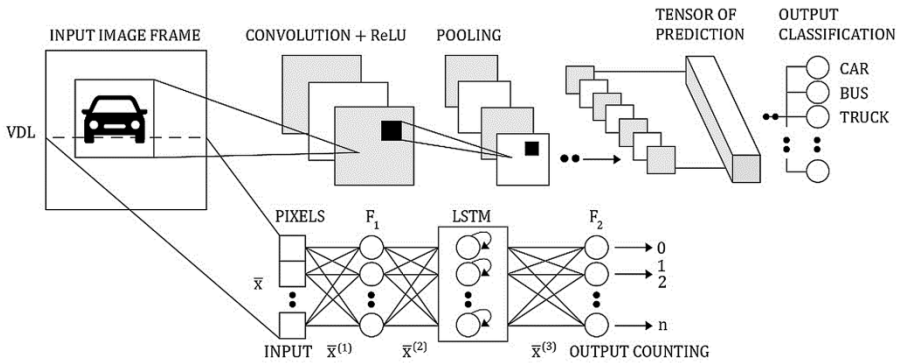

Fig. 3. Topology of CNN and RNN-LSMT assemble for the automatic data labeling and counting.

The first layer of the network is a fully connected layer which is a vector consisting of a line of pixels y_l. In our experiment there were 100 neurons plus bias. For the activation function of the fully connected layer we used ReLU. This means we have got the output a vector $h^{(t)} = ReLU(W_{ih}(x^{(t)} + b_h)$ after the first fully connected layer. This

vector has been forwarded to the LSTM layer. The number of neurons is the same 100. The LSTM internal memory allows the network to identify if there is an object or objects on the VDL. Additionally, it gives an impression the direction in which the line is crossed and whether there are some objects on the detection line. The output layer of the RNN-LSTM network is a fully connected layer with a softmax function. In our experiment the number of output neurons corresponded to 10. This number represents 10 possible output classes for each input vector $x^{(t)}$. Sparse coding has shown how many vehicles are on the VDL. The proposed object detection and data labeling methods were implemented in Python 3. The TensorFlow framework was used for implementation and training of the RNN-LSTM network.

7. Conclusions

The main contribution is that we have solved the problem of how to smartly and efficiently generate data labels automatically for one neural network using another neural network. The proposed approach has been tested using two different neural network architectures: the convolution neural network and the recurrent neural network with long short-term memory cells. The economy and efficiency of this system comes from using one virtual detector line instead of the whole frame, which makes the proposed approach scalable for such applications as Intelligent Transportation Systems. As shown by the test, this approach can substantially decrease the training time for the RNN-LSMT network. There is no need for manual data labeling and does not involve the human resources.

Further research would be consisting of investigating the accuracy of the proposed automatic data labeling when the detection line is quite close to the upper and lower areas of the bounding box. Another area of research would be the use of smoothing filters for automatic data labeling. In the future, YOLO can be replaced by more precise networks that already exist to further improve the quality of training data. The proposed labeling method allows researchers or practitioners to obtain the necessary training and testing data much faster compared with the manual or semi-manual labeling methods. Furthermore, it can be used get data labels for other tasks, for example, counting or inspecting objects in robotics.

References

[1] Gross, Michael. (2016) “A planet with two billion cars.” *Current Biology Magazine*, **26**:R307-318.

[2] Mithun, Niluthpol, Nafi Rashid, and S.M. Mahbuban Rahman. (2012) “Detection and Classifiction of Vehicle From Video Using Muliple Time-Spatial Images”, *IEEE Transactions on intelligent transportation systems*, **13 (3)**: 1215-1225.

[3] Redmon, Joseph, and Ali Farhadi. (2016) “YOLO9000: Better, Faster, Stronger,”: arXiv:1612.08242.

[4] Lin, Tie, Sun Jian, Tang Xiaoon, and Shum Heung-Yeung. (2007) “Learning to Detect A Salient Object”, IEEE.

[5] Chandran, Ranjeethkuman, and Naveen Raman. (2017) “A Review on Video-Based Techniques for Vehicle Detection, Tracking and Behavior Understanding”, *International Jourmal of Advances in Computer and Electronics Engineering*, **2 (5)**: 07-13.

[6] Deng, Zhipeng, Sun Hao, Zhou Shilin, Lei Lin, and Zou Huanxin. (2018) “Multi-scale object detection in remote sensing imagery with convolutional neural networks”, *ISPRS Journal of Photogrammetry and Remote Sensing*, **145**: 3-22.

[7] Suhaoo, Li, Lin Jinzhao, Li Guoquan, Tong Bai, Huiqian Wang, and Yu Pang. (2018) “Vehicle type detection based on deep learning in traffic scene”, *Procedia Computer Science*, **131**: 564-572.

[8] Woonhyun, Nam, and Joonhee Han. (2006) “Motion-based background modelling for foreground segmenttion”, *VSSN`06: Proceedings of the 4th ACM international workshop on Video surveillance and sensor neworks*,: 35-44.

[9] Sauer, Ken, and Collen Jones. (1993) “Bayesian Block-Wise Segmentation of Interframe Differences Video Sequences”, *CVGIP: Graphical Models and Image Proceeding*, **55(2)**: 129-133.

[10] Kalai, Adam Tuman, Ankur Moitra, and Gregory Valiant. (2010) “Efficiently learning mixtures of Gaussians”, *STOC`10: Proceedings of the forty-second ACM symposium on Theory of computing*,: 553-562.

[11] Kuo, Yin-Che, Pai Ning-Sheng, and Li Yen-Feng. (2011) “Vision-based vehicle detection for a driver assistance system”, *Computer and Mathematics with Applications*, **61**: 2096-2100.

[12] Tong, Hongshen, Ni Rongrong, Zhao Yao, and Li Xiaolong. (2018) “Median filtering detetion of small-size based on CNN”, *Journal of Image Representation*, **51**: 162-168.

[13] Hamuda, Esmael, Brian Mc Ginley, Marti Glavin, and Edward Jones. (2018) “Improved image-based crop detection using Kalman filtering and the Hungarian algorithm”, *Computers and Electronics in Agriculture*, **148**: 37-44.

[14] Sharma, Kajal. (2018) “Feaure-based efficient vehilce tracking for a traffic surveillance system” *Computers & Electrical Engineering*, **70**: 690-701.

[15] Li, Yali, Wang Shengjin, Tian Qi, and Ding Xiaoqing. (2014) “Learning cascaded shared-boot classifiers for part-based object detection”, *IEEE Transactions on Image Processing: A Publication of the IEEE Signal Processing Society*, **23(4)**: 1858-1871.

[16] Lowe, David. (2004) “Distintive Image Feature from Scale-Invariant Keypoints”, *International Journal of Computer Vision*, **60**: 91-100.

[17] Bay, Herbert, Tuytelaars Timme, and Luc van Gool. (2006) *European Conference on Computing Vision*, **1**: 404-417.

[18] Mikolajczyk, K., and C. Schmid. (2005) “A performance evaluation of local descriptors”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **29**: 1615-1630.

[19] Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. (2014) “Rich feature hierachies for accurate object and semantic segmentation. Tech report(v5), arXiv: 1311.2524v5

[20] Girshick, Ross. (2015) “Fast R-CNN”, arXiv:1504.08083v2. Available: https://github.com/rbgirshick/fast-rcnn. [Accessed : September 18, 2018].

[21] Ren, Shaoqing, He Kaiming, Girshick Ross, and Sun Jian. (2016) “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, arXiv:1506.01497v3

[22] He, Kaiming, Gkioxari, Georgia Dollar, Piotr, and Ross Girshick. (2018) “Mask R-CNN”, arXiv:1703.06870v3

[23] Shen, Zhiqiang, Liu Zhuang, Li Jianguo, Jiang Yu-Gang, Chen Yurong, and Xue Xiangyang. (2018) “Object Detection from Scratch with Deep Supervision”, arXiv: 1809.09294v1

[24] Redmon, Joseph, Divvala Santosh, Girshick Ross, and Farhadi Ali. (2016) “You Only Look Once: Unified, Real-Time Object Detection”, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*: 779-788.

[25] Redmon, Joseph, and Farahdi Ali. (2018) “YOLOv3: An Incremental Improvement”. Available: https://pjreddie.com/media/files/papers/YOLOv3.pdf. [Accessed: October 26, 2018].

[26] Mhalla, Ala, Chateau Thierry, Maamatou Honda, Gazzah Sami, and Ben Najoua. (2017) “SMC faster R-CNN: Toward a scene-specialized multi-object detector”, *Computer Vision and Image Understanding*, **164**: 3-15.

[27] Graves, Alex, and Jaitly Navdeep. (2014) “Towrads End-to-End Speech Recogniton with Recurrent Neural Networks”, *Prceedings of the 31st International Conference on Machine Learning*, **14**: 1764-1772.

[28] Martinez, Julieta, J. Michael Block, and Romero Julieta. (2017) “On human motion prediction using recurrent neural networks”, *In Computer Vision Foundation (CVPR 2017)*, : 2891-2900.

[29] Namatevs, Ivars. (2017) “Deep Convolutional Networks: Structure, Feature Extraction and Training”, *Journal of Information Technology and Management Science*, **20**: 40-47.

[30] Melekhov, Jaroslav, Ylioinas Juha, Kannala Juho, and Rahtu Esa. (2017) “Relative Camera Pose Estimation Using Convolutional Neural Networks”, arXiv:1702.01381v3

[31] Russakovsky, Olga, Deng Jia, Su, Hao Krause, and Jonathan et.al. (2015) “ImageNet Large Scale Visual Recognition Challenge”, *Internatioal Journal of Copmputer Vision*, **115(3)**: 211-252.

[32] Lin, Tsung-Yi., Maire Michael, Belongie Serge, and Bourdev Lubomir et al. (2015) “Microsoft COCO: Common Object in Context: arXiv:1405.0312v3

[33] Nanni, Loris, Brahnam Sheryl, Ghidoni Stefano, and Menegatti Emanuele. (2015) “Improving the descriptores extracted from the co-occurrence matrix using preprocessing approaches”, *Expert Systems With Appliction*, **42**: 8989-9000.

[34] Graves, Alex. (2014) “Generating Sequence With Recurrent Neural Networks”, arXiv:1308.0850v5

[35] Boden, Mikale. (2001) “A guide to recurrent neural networks and backpropagation”, *School of Information Science, Computer and Electrical Engineering, Halmstad University*: 1-10.

[36] Goodfelow, Ian, Bengio Yoshua, and Courville Aaron. (2016) “Deep Learning (Adaptive Competition and Machine Learning), The MIT Press.

[37] Hochreiter, Sepp, and Schmidhuber Jürgen. (1997) “Long Short-Term Memory”, *Neural Computation*, **9(8)**:1735-1780.

[38] Kadikis, Roberts. (2018) “Recurrent Neural Network Based Virtual Detection Line”, *The 10th International Conference on Machine Vision (ICMV 2017)*.

[39] Valev, Krassimir, Schuman Arne, Sommer Lars, and Beyerer Jürgen. (2018) “A Systematic Evaluation of Recent Deep Learning Architectures for Fine-Grained Vehicle Classification”, arXiv:1806.02987v1

Ivars Namatevs holds a *Mg. sc. ing.* Automation and Telemechanic from Riga Technical University and an MBA degree from Riga Business School. His research interests include deep artificial intelligence, especially application of neural networks for unsupervised and reinforcement learning in machine learning, as well as foresight for artificial intelligence. Contact information: 68, Graudu Street, Riga, LV-1058, Latvia; phone: +371 26133567. Contact him at ivars@turiba.lv.