Pedestrian detection in video surveillance using fully convolutional YOLO neural network

Molchanov, V., Vishnyakov, B., Vizilter, Y., Vishnyakova, O., Knyaz, V.

V. V. Molchanov, B. V. Vishnyakov, Y. V. Vizilter, O. V. Vishnyakova, V. A. Knyaz, "Pedestrian detection in video surveillance using fully convolutional YOLO neural network
," Proc. SPIE 10334, Automated Visual Inspection and Machine Vision II, 103340Q (26 June 2017); doi: 10.1117/12.2270326

Event: SPIE Optical Metrology, 2017, Munich, Germany

Pedestrian detection in video surveillance using fully convolutional YOLO neural network

V. V. Molchanov[a], B. V. Vishnyakov[a], Y. V. Vizilter[a], O. V. Vishnyakova[b], and V. A. Knyaz[a]

[a]The Federal State Unitary Enterprise ”State Research Institute of Aviation System”,
Viktorenko street, 7, Moscow, Russia
[b]Higher School of Economics, Myasnitskaya, 20, Moscow, Russia

ABSTRACT

More than 80% of video surveillance systems are used for monitoring people. Old human detection algorithms, based on background and foreground modelling, could not even deal with a group of people, to say nothing of a crowd. Recent robust and highly effective pedestrian detection algorithms are a new milestone of video surveillance systems. Based on modern approaches in deep learning, these algorithms produce very discriminative features that can be used for getting robust inference in real visual scenes. They deal with such tasks as distinguishing different persons in a group, overcome problem with sufficient enclosures of human bodies by the foreground, detect various poses of people. In our work we use a new approach which enables to combine detection and classification tasks into one challenge using convolution neural networks. As a start point we choose YOLO CNN, whose authors propose a very efficient way of combining mentioned above tasks by learning a single neural network. This approach showed competitive results with state-of-the-art models such as FAST R-CNN, significantly overcoming them in speed, which allows us to apply it in real time video surveillance and other video monitoring systems. Despite all advantages it suffers from some known drawbacks, related to the fully-connected layers that obstruct applying the CNN to images with different resolution. Also it limits the ability to distinguish small close human figures in groups which is crucial for our tasks since we work with rather low quality images which often include dense small groups of people. In this work we gradually change network architecture to overcome mentioned above problems, train it on a complex pedestrian dataset and finally get the CNN detecting small pedestrians in real scenes.

Keywords: Video surveillance, pedestrian detection, object classification, convolutional neural networks, human detection

1. INTRODUCTION

Nowadays video surveillance is mostly used for monitoring people indoors and outdoors. As example of using these methods one can suggest different security video systems which aims to detect violators in restricted areas, video monitoring in airports, public spaces, working places, offices etc. So, the problem of efficient human monitoring can be applied to a number of different tasks. However, in all these applications we have to face problems connected with bad image quality, complex illumination conditions, PTZ cameras, sufficient enclosures of objects by the foreground, strict conditions imposed on computational and time resources given for particular algorithm. This restrictions influence algorithm's quality as negative factors, reducing it's efficiency.

Over the last decade a number of approaches for people detection, based on pattern recognition, was proposed (more than 1000, including the approaches and their modifications [1], [2]). Comparative testing and brief overview of the key components of them can be found in [3]. Also a lot of background modelling approaches as Gaussian mixture models [4], [5], kernel density estimation [6], eigen background [7], codebook [8] and etc. [9], [10].

Further author information: (Send correspondence to B. V. Vishnyakov)
V. V. Molchanov: E-mail: vmolchanov@gosniias.ru,
B. V. Vishnyakov: E-mail: vishnyakov@gosniias.ru,
Y. V. Vizilter: E-mail: viz@gosniias.ru,
O. V. Vishnyakova: E-mail: vishnyakova.oxana.pro@gmail.com,
V. V. Knyaz: E-mail: knyaz@gosniias.ru

Automated Visual Inspection and Machine Vision II, edited by Jürgen Beyerer,
Fernando Puente León, Proc. of SPIE Vol. 10334, 103340Q · © 2017 SPIE
CCC code: 0277-786X/17/$18 · doi: 10.1117/12.2270326

These approaches provide a clustering of moving foreground regions that can be afterwards classified as humans, cars, animals etc. [11], [12], [13]. But nowadays it is obvious that the methods, based on the convolutional neural networks, provide much better results than the other ones in terms of detection quality.

In the first papers on neural networks for pedestrian detection authors used prior detection systems that repurpose classifiers or localizers to perform detection. They apply the model to an image at multiple locations and scales. Such strategy was exploited in paper which proposed rather complicated algorithm, which included three successive stages and called R-CNN [14]. At the first stage authors generate a large number of regions of interest, which probably could contain some objects or their parts. At the next stage convolution neural network was applied to all proposed regions, pre-trained on a classification task to get discriminative features. And then at the final stage authors used SVM classifiers, based on these features, to make a final decision. This approach has some evident drawbacks, among which are slow speed that prevent using it in real-time systems and also too complicated architecture, that couldn't be learned as a whole one. To eliminate these shortcomings, several different neural networks with similar architecture were proposed, that combine separate stages of previous works into one CNN and converge this task to the regression problem, which enables to learn this networks as a whole from the beginning to the end. Two of such CNNs are YOLO [15] and DetectNet [16]. We will conduct research on YOLO, but DetectNet is very similar. Strictly speaking YOLO is not a single net but rather a way of building networks for detection purposes. Authors use common CNN which is pre-trained on classification tasks and then add some fully-connected layers on top, which are trained on detection task. In the paper authors aim to predict bounding boxes for each object on an image and corresponding class labels for them. To accomplish that they divide image on regular 7x7 grid and for each grid cell they predict: object bounding boxes sizes and locations relative to the center of the cell, probabilities scores per class. Loss function include both b-boxes term and probabilistic term, so classification and detection tasks are being resolved simultaneously. Thanks to applying CNN to the whole image rather than to its parts, such nets make their predictions using global context and, as a result, making less mistakes. Also these nets show relatively high performance in real time challenges.

Despite all advantages YOLO suffers from some know drawbacks related to the fully-connected layers, that obstruct applying it to images with different resolution. Also it limits the ability to distinguish small close human figures in groups which is crucial for our tasks since we work with rather low quality images which often include dense small groups of people. First problem is obvious: fully-connected layers require fixed input size and this requirement propagates to the input image, so we can't apply YOLO CNN to an arbitrary image without resizing it to the dimensions, defined by net. Resizing can lead to geometry deformations which could be harmful, and it also decreases object sizes on the edges, which relates to problem with small objects. Also there is a way of cropping an image into pieces of the fixed size, but it turns into a clustering problem on the edges of the cropped and resized images.

Challenge with small objects has two sources. At first, YOLO use models that are pre-trained on data, sets such as ImageNet [17], and having top layers fine-tuned on PascalVOC [18] or Kitti [19], which don't contain small objects. Secondly, YOLO predicts fixed number of objects per cell (2 in original paper), so if there is a group of people whose size is very small compared to the cell size, then most of them would be cut off.

In this paper we use the following approaches to solve the mentioned above problems. At first we resolve the issue with the input image size by replacing fully-connected layers with convolutions, what could be made without retraining for original image sizes (but not always, as we show in Section 2) since fully-connected layers can be treated as convolutions. So, the whole net transforms into a convolutional one and could be used as a sliding window for images with bigger sizes. Then we address the issue connected with small object sizes, we first fine-tune the CNN on our datasets, containing small objects on low quality images, we also use grid with cell sizes comparable with sizes of the smallest objects we want to detect - such approach is also used in DetectNet. As a result we get efficient network architecture for robust pedestrian detection in video surveillance.

The remainder of this paper is organized as follows. In Section 2 we describe our approach, addressing the problem of the fixed input size. Section 3 is dedicated to issue with small object sizes, Section 4 describes experimental results and Section 5 concludes this paper.

2. ADAPTATION OF YOLO CNN TO ARBITRARY IMAGE SIZES

As has been said, the way of adaptation of YOLO CNN to arbitrary sizes is based on the idea of substitution of fully-connected layers for convolutional layers [20]. Below we describe this method in details.

When we submit an image to the neural network input it passes through all its layers until first fully-connected layer, let's denote its dimension as c_{out} and let input feature map for this layer has dimensions w, h, c_{in}. Fully-connected layer treats this feature map as a one-dimensional array of size $w \cdot h \cdot c_{in}$. So, weight of this layer forms matrix $W_{c_{out}, w \cdot h \cdot c_{in}}$ and each neuron $i = 1 \ldots c_{out}$ from this layer has $w \cdot h \cdot c_{in}$ connections with neurons from the previous layer. Hence, if we threat this layer as a convolutional one and it's neurons as filters of sizes $w \cdot h \cdot c_{in}$, then it helps easily face the problem, caused by fully-connected layers. We apply this approach to different network architectures that would be described in more detains further.

In simple case, when we use CNN for images of it's original size - making a convolutional layer out of a fully-connected layer is the only thing that we have to do. Indeed, to convolve input feature map with filter we transform both as shown in (Fig. 1) and then perform simple matrix multiplication with successive transformation to get the correct shape as a result. So we simply can initialise filter's weights in convolution layers by values of weights from fully-connected layers, and due to nature of convolutional operation we get the same result if we transform it to one-dimensional array.

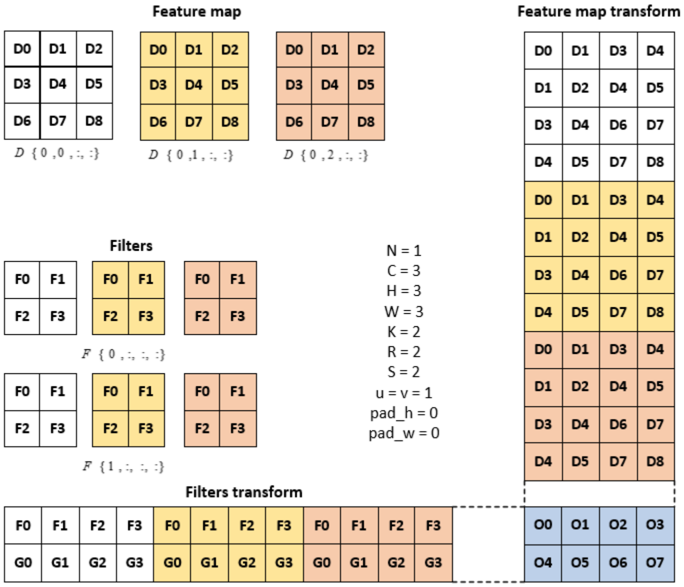


Figure 1. Representing convolution operation as a matrix multiplication.

However, it's not valid for images of arbitrary sizes, that can be explained by existence of non zero padding in many layers. To show this, consider next feature map as an input to particular convolutional layer (Fig. 2). For simplicity let's assume that there is only one filter with sizes 3 × 3 and feature map has one channel, other layer parameters: $stride = 1, pad = 1$. Together with this feature map we consider another one, which is obtained from the first one with additional row and additional column (this happens when we have large image and then cut some part from it).

Convolution result of both these feature maps is present on (Fig. 3) below. After achieving such results we can conclude that, despite similarity of some parts (marked yellow) in resulted maps, they are not equal due to side effects caused by non-zero padding. This effect increases with the network depth. So, by contrast to simple case when an input image has suitable sizes or zero padding it could be harmful to use weights from fully-connected layers especially for too large image sizes, thereby new convolution layers should be retrained.

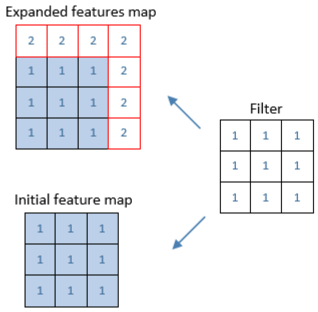


Figure 2. Expanded feature map (4 × 4) with initial feature map (3 × 3) and filter (3 × 3).

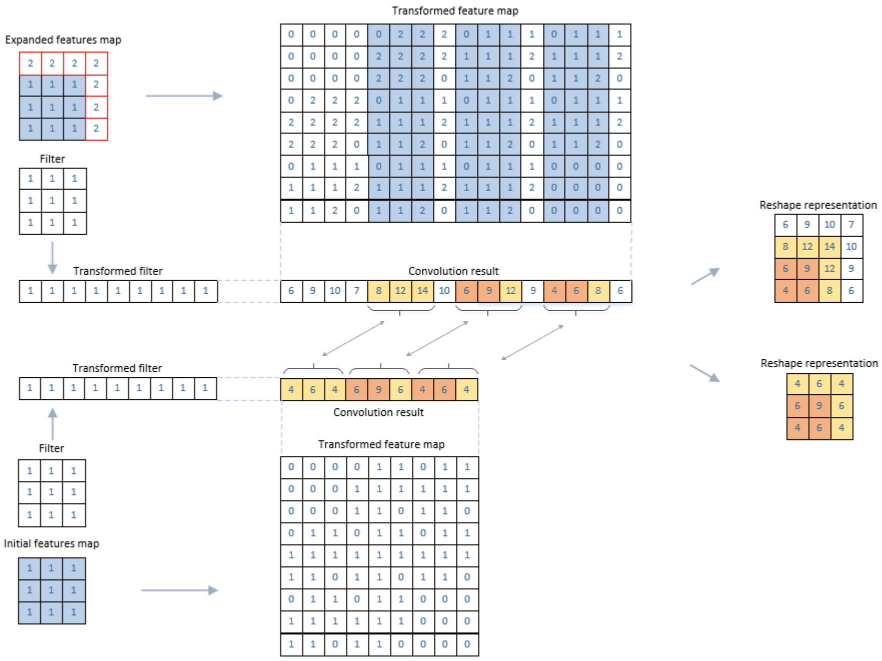


Figure 3. Convolution result for both feature maps. Boxes with red borders affect the calculation result due to a non-zero padding. We marked orange outputs that are coincide for both maps and yellow that are not.

3. SMALL OBJECT SIZES

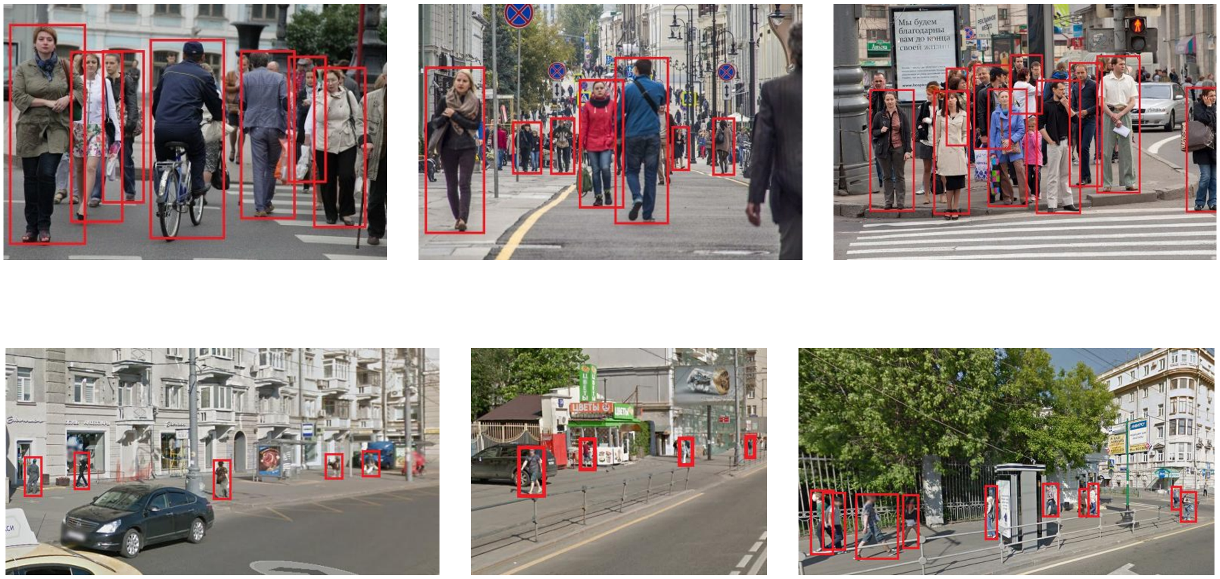
As we mentioned above in Introduction, YOLO CNN has problems with small objects. The reason is that YOLO uses models pre-trained on datasets, such as ImageNet, and then fine-tuned (only top layers) on PascalVOC or KITTY, which do not contain small objects. Also, YOLO predicts fixed number of objects per cell (2 in original paper), so if there is a group of people whose size is very small compared to the cell size, then most of them would be cut off. To address this drawback we primarily retrain new convolutional layers on our datasets, and also decrease cells sizes to be equal to the smallest object sizes we want to detect.

After network convolutionalization we have two-dimensional array of predictions, each column of which contains network outputs of the same type like in original YOLO-tiny CNN (the shortest version of YOLO CNN). For example, original YOLO architecture was trained on images of size 448×448×3. Output from last convolution layer has size 7×7×1024. First fully-connected layer has 256 neurons, so we get the output size 256. If we use image size 640×640×3 as an input to our convolution version of YOLO-tiny CNN, then the same convolutional layer gives us the output size 11×11×1024, instead of fully-connected one, that gives us output size 5×5×256.

We then combine these predictions using clustering algorithm called ”groupRectangles” from OpenCV. Instead of the original way, suggested by authors of YOLO CNN, where intersection over union measure (IOU) is used to find the best predicted box for one in ground truth, it uses a special metric to determine measure of similarity between rectangles. Then it forms clusters based on this measure. After that, average rectangle is calculated for each cluster and announced as a prediction result. Algorithm has some configuration parameters: minimal number of rectangles in cluster, metric's sensitivity measure (in extreme cases all rectangles follows up in one cluster or each rectangle forms separate cluster).

4. EXPERIMENTS

In our experiments we consider three original neural network architectures arranged in decreasing power order: YOLO, YOLO-small and YOLO-tiny. The first one possesses the best quality characteristics and could compete with the state-of-the-art object detection models. But it consumes too many resources for our particular tasks, so for experiments we focused on YOLO-tiny architecture, which is much faster and suits our computation resources budget. This model attains the same top-1 and top-5 performance as AlexNet [21], but with ten times less parameters. It uses mostly convolutional layers without the large fully-connected layers. It is about twice as fast as AlexNet on CPU. We change it's fully-connected layers with the new convolutional ones and retrain this model on images from our own dataset, that contains people of small sizes, starting with 12 pixels wide. This dataset is a mix of Caltech, KITTY datasets with our own data, gathered from video sequences of Moscow city surveillance system, which contain about 1000 images with 10-15 persons per image in average. The train process was divided on two stages: at first we pre-train the new convolution layers on Caltech [22] and KITTY datasets, and next we fine-tune it on our own datasets which contain pedestrians of smaller sizes and include lots of scenes with bad image quality. During first stage we do not totally freeze first layer. We allow them to adapt to the new data, however we restrict their learning rates by factor 0,1 in comparison with the new layers. At the final stage we perform learning only for newly added layers. On both stages we used stochastic gradient descent RMSProp (unpublished, adaptive learning rate method proposed by Geoff Hinton) with exponential decay schedule, started from 0,1 learning rate. We also use Batch Normalisation, since first layers were trained using it. We choose Caffe for our experiments and implementation of YOLO new layers. Figures (Fig. 4) introduce experimental results on mixed datasets which contains images of different sizes.



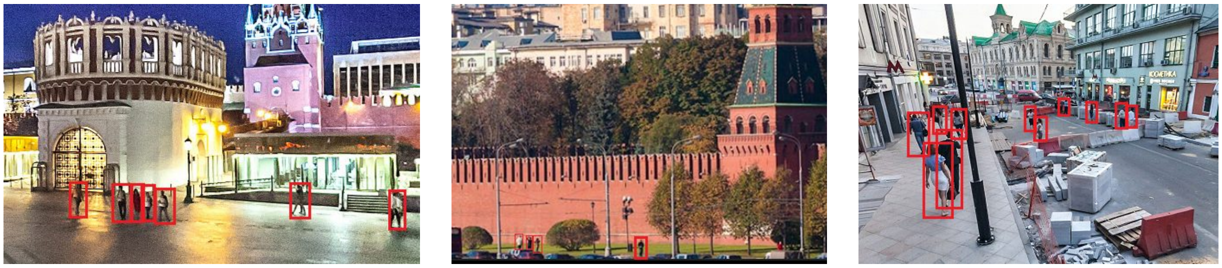


Figure 4. Examples of the real images from different datasets.

5. CONCLUSION

In this paper, we apply method for object detection with single convolution neural network to pedestrian detection tasks in video surveillance. We adopt it to images with arbitrary sizes that prevents us from retraining network for each particular size and helps to avoid image resizing which leads to geometry deformation and decreasing of detection quality. We changed fully-connected layers with convolutional ones and trained them from scratch. Then we addressed a problem, connected with presence of small persons on an input image. First of all we train models on our task specific datasets with small persons and decrease grid cell size, making it approximately equal to the smallest object size we want to detect. We also changed algorithm for clustering of final bounding boxes, by splitting all bulk of predictions on clusters and getting average rectangle from each cluster as final conclusion.

ACKNOWLEDGMENTS

This work was supported by Russian Science Foundation (RSF) under Grant 16-11-00082; by Russian Foundation For Basic Research (RFBR) under Grants 15-07-09362.

REFERENCES

[1] Zhang, S., Benenson, R., and Schiele, B., "Filtered channel features for pedestrian detection," *CoRR* **abs/1501.05759** (2015).

[2] Benenson, R., Omran, M., Hosang, J. H., and Schiele, B., "Ten years of pedestrian detection, what have we learned?," *CoRR* **abs/1411.4304** (2014).

[3] Viola, P., Jones, M. J., and Snow, D., "Detecting pedestrians using patterns of motion and appearance," *International Journal of Computer Vision* **63**(2), 153–161 (2005).

[4] Stauffer, C. and Grimson, W. E. L., "Adaptive background mixture models for real-time tracking," in [*CVPR*], 2246–2252, IEEE Computer Society (1999).

[5] Zivkovic, Z., "Improved adaptive gaussian mixture model for background subtraction," in [*Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*], **2**, 28–31 Vol.2 (2004).

[6] Elgammal, A., Duraiswami, R., Harwood, D., and Davis, L. S., "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proceedings of the IEEE* **90**, 1151–1163 (Jul 2002).

[7] Rymel, J., Renno, J., Greenhill, D., Orwell, J., and Jones, G. A., "Adaptive eigen-backgrounds for object detection," in [*Image Processing, 2004. ICIP '04. 2004 International Conference on*], **3**, 1847–1850 Vol. 3 (Oct 2004).

[8] Kim, K., Chalidabhongse, T. H., Harwood, D., and Davis, L., "Background modeling and subtraction by codebook construction," in [*Image Processing, 2004. ICIP '04. 2004 International Conference on*], **5**, 3061–3064 Vol. 5 (Oct 2004).

[9] Goyette, N., Jodoin, P. M., Porikli, F., Konrad, J., and Ishwar, P., "Changedetection.net: A new change detection benchmark dataset," in [*2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*], 1–8 (June 2012).

[10] Sidyakin, S. V., Vishnyakov, B. V., Vizilter, Y. V., and Roslov, N. I., "Mutual comparative filtering for change detection in videos with unstable illumination conditions," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **XLI-B3**, 535–541 (2016).

[11] Omar Javed, M. S., "Tracking and object classification for automated surveillance," *ECCV '02 Proceedings of the 7th European Conference on Computer Vision-Part IV* , 343–357 (2002).

[12] Brown, L. M., "View independent vehicle/person classification," *VSSN '04 Proceedings of the ACM 2nd international workshop on Video surveillance and sensor networks* , 114–123 (2004).

[13] Jiman Kim, D. K., "Fast car/human classification using triple directional edge property and local relations," *2009 11th IEEE International Symposium on Multimedia* , 106–111 (2009).

[14] Ross B. Girshick, Jeff Donahue, T. D. J. M., "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR* **abs/1311.2524** (2013).

[15] Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi, A., "You only look once: Unified, real-time object detection," *CoRR* **abs/1506.02640** (2015).

[16] Tao, A. and Barker, J., "Detectnet: Deep neural network for object detection in digits." https://devblogs.nvidia.com/parallelforall/detectnet-deep-neural-network-object-detection-digits/.

[17] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L., "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)* **115**(3), 211–252 (2015).

[18] Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A., "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision* **88**, 303–338 (June 2010).

[19] Geiger, A., Lenz, P., and Urtasun, R., "Are we ready for autonomous driving? the kitti vision benchmark suite," in [*Conference on Computer Vision and Pattern Recognition (CVPR)*], (2012).

[20] Shelhamer, E., Long, J., and Darrell, T., "Fully convolutional networks for semantic segmentation," *CoRR* **abs/1605.06211** (2016).

[21] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," in [*Advances in Neural Information Processing Systems 25*], Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., eds., 1097–1105, Curran Associates, Inc. (2012).

[22] Dollár, P., Wojek, C., Schiele, B., and Perona, P., "Pedestrian detection: An evaluation of the state of the art," *PAMI* **34** (2012).