Object detection algorithm for high resolution images based on convolutional neural network and multiscale processing

Rykhard Bohush[a], Sergey Ablameyko[b,c], Sviatlana Ihnatsyeva[a] and Yahor Adamovskiy [a]

[a] *Polotsk State University, 29 Blokhin st., Novopolotsk, 211440, Belarus*
[b] *Belarusian State University, 4 Nezavisimosti Avenue, Minsk, 220030, Belarus*
[c] *United Institute for Informatics Problems, National Academy of Sciences of Belarus, 6 Surganova st., Minsk, 220012, Belarus*

Abstract

In this article we propose an effective algorithm for small object detection in high resolution images. We look at the image at different scales and use block processing by convolutional neural network. Pyramid layers number is defined by input image resolution and convolutional layer size. On each layer of pyramid except the highest we perform splitting overlapping blocks to improve small object detection accuracy. Detected areas are merged into one if they belong to the same class and have high overlapping value. In the paper experimental results using YOLOv4 for 4K and 8K images are presented. Our algorithm shows better detecting small objects results in high-definition video than YOLOv4.

Keywords 1

Small objects detection, Image pyramid, Convolutional neural networks, YOLO.

1. Introduction

Video surveillance systems are used to solve various problems such as identification of pedestrians, vehicles, car numbers detection, in security system and other. Now day 4K and 8K camcorders are able of recording high-definition video, which can identify object details to improve and increase the possibility of detection tiny objects [1]. These notions refer to any digital images containing resolution of approximately 4000 or 8000 pixels [2]. The advantage of using high resolution systems is the ability not only to accurately small objects detect but also to display the correct large objects shape. It allows to improve detection accuracy. However, there is also a negative sides, the main one of which is a large information capacity in each frame or image. This leads to the need to use fast and efficient image processing algorithms to process high resolution images and video in real time.

Convolutional neural networks (CNNs) are very effective for object detection and became widespread due to the rapid growth in computing power. Convolutional layers with different filter types allow to create effective feature sets to describe images that have unclear structure and many variations within a class. However, GPU computing power and memory size is still insufficient for processing high-resolution images. Therefore, when using a CNN, input image is scaled to the input layer size at the first stage. After that, input image is less informative for further processing. It is important to take into account that image resolution increasing reduces percentage of object size if it is constant in pixels. At present, most known systems do not consider input image resolution. Such systems scale to the input layer size and if image or video frame with high-resolution, tiny objects can

CMIS-2021: The Fourth International Workshop on Computer Modeling and Intelligent Systems, April 27, 2021, Zaporizhzhia, Ukraine
EMAIL: r.bogush@psu.by (R. Bohush); ablameyko@bsu.by (S. Ablameyko); s.ignatieva@psu.by (S. Ihnatsyeva); e.adamovsky@psu.by (Y. Adamovskiy)
ORCID: 0000-0002-6609-5810 (R. Bohush); 0000-0001-9404-1206 (S. Ablameyko); 0000-0002-9780-5731 (S. Ihnatsyeva); 0000-0003-1044-8741 (Y. Adamovskiy)

be missed. Consequently, the algorithms development for detecting small objects in high-resolution images is an urgent task.

This article is structured as follows. In Section 2, related algorithms based on CNNs for object detection in images are illustrated. In Section 3 our algorithm by multiscale and block processing is presented. In Section 4 experimental results using database of 4K images and discussion are shown. Finally, the conclusion and feature work details are provided in Section 5.

2. Related Works

Generally, most of the image object detection algorithms for high definition images are carried out using two principals: image dividing into blocks and CNNs application for each block. In [3], the algorithm for the accurate of small objects detection in high-definition video are presented. Each image is separated into overlapping blocks and object detection in each block is performed with CNN YOLO. After that, the detected objects post-processing is performed in each subframe and the same class probabilities adjacent areas are marge. Compared to the classic YOLO, the algorithm shows the better results when detecting tiny objects in high-definition video. But the main disadvantage of this algorithm is sufficiently high probability of combining several different objects that are located at the processed blocks boundaries into one region and skipping the object due to its fragmentation into blocks.

Yongxi and Javidi [4] use neighbor visual features and high-level image features to predict bounding boxes to guide a two-stage search process that adaptively focuses on areas that may contain tiny objects. Objects are detected on a reduced image copy using the AlexNet model. The original image was divided into non-overlapping blocks to detect small objects. Sizes of such blocks varied from 600 to 75 pixels and blocks were processed using the SCN (Spatial Correlation Network). The overlap varied from 25 to 50% relative to the block sizes. For this algorithm, authors obtained a recall value of 80% based on SUN2012 dataset [5]. It is known that the recall metric doesn't allow evaluating false positives, and the dataset used by the authors contained almost no images with 4K resolution.

In [6] the algorithm for detecting human uses a two-stage approach. First, the original image is scaled down to low-resolution space. Low resolution object detection guides the attention of the model to the important areas of the input image. In the second step, the algorithm is returned to high resolution only, only reviewing the selected areas. Large blocks are split into smaller blocks to refine the bounding boxes of the detected regions of interest. Found regions merging is based on the vertically elongated (rectangular) person shape. This means that the algorithm is not adapted for other object classes. For both stages model YOLO v2 is applied. Algorithm accuracy was estimated using PEVID-UHD dataset [7] and a closed dataset containing a slightly larger number of objects. Average precision is 90.7% for the first dataset and 75.4% for the second. At the same time, in some cases, the use of block overlap in the range of 20 to 50 pixels is not sufficient; in addition, it is not adaptive to input image size.

Unel et al. [8] use CNN PeleeNet for detecting small objects of the "vehicle" and "pedestrian" classes in images with 2K resolution and dimensions [1920×1080] onboard a micro aerial vehicle.

The algorithm uses block image processing of the PeleeNet model with 25% overlap between the tiles. For this algorithm mAP metric on VisDrone2018 dataset is 36.67%. The proposed technique is implemented on Nvidia Jetson TX1 and TX2. However, this paper does not provide an estimate of the block overlap effect on the algorithm accuracy. The number of missed objects of the "pedestrian" class is significantly higher than the "vehicle" class. Consequently, this algorithm detects small objects much worse even for images with a resolution below 4K.

As we can see, it is important to use CNN with high accuracy but with low computational costs. CNN YOLO favorably differentiate by other models because it processes entire image without previous RoI extraction. It means that CNN considers contextual information about whole image and it's important. YOLO is a one-stage object detector. In addition, as stated in [9] YOLO is the fast object detection system that performs better than famous Faster R-CNN. This is of great importance because computational costs is a significant parameter when processing high-resolution image and video frames. YOLO is more effective by computational cost. On the downside, there are many

localization errors particularly for tiny objects, due to the input image scaling. At paper [10], considers YOLO modifications: YOLO v2, YOLO9000. Improvement in segmentation an classification has been achieved through the use of new Darknet-19 classification model with anchor boxes; batch normalization from [10]; using k-means; direct location prediction; RPN usage; fine-grained features; multi-scale training. Top-5 accuracy was 91.2% but proposed algorithm took a big computational cost. Also multi-scale training gave better opportunity to detect small objects, but it is not enough for 4K resolution systems. YOLOv2 [10] uses five anchor-boxes, CNN Darknet-19, and Batch Normalization for predictions on each convolutional layer, which increased the detection accuracy. The next version as YOLOv3 [11] has further improved its detection accuracy indicators. This was achieved using CNN darknet-53, residual connection, multi-scale object detection through the FPN (Feature Pyramid Network) use. YOLOv3 uses 1×1 detection kernels on feature maps for object detection in three different network layers for three different sizes. This improves detection accuracy and detects smaller objects. However, YOLOv3 loses out in terms of speed compared to YOLOv2. A solution to this problem is proposed in YOLOv4.

YOLOv4 represents state-of-the-art architecture, with one of the best ratio accuracies and detection object speed [12]. The approach is based on the Darknet53 convolutional neural network that uses Cross-Stage-Partial-connections (CSP), pyramids for feature extraction, anchor-box for detection. To achieve high detection speed rates and accuracy, as well as the ability to use available training equipment, YOLOv4 uses several methods to increase efficiency without computational cost or with negligible computational complexity. These include: using genetic algorithms to optimize the learning rate, Mosaic and CutMix dataset augmentation methods, SAT (Self-Adversarial Training), modified SAM (Spatial Attention Module) and PAN (Path aggregation Network) modules, SPP (Spatial Piramid Pooling), class label smoothing, CmBN - using Cross mini-Batch Normalization, Mish activation, CIoU-loss, etc. Various optimization methods combinations allow to achieve an average precision AP of about 43%. Thus, as a model of CNN for object detection in high-resolution images, we will use YOLO v4.

3. Object detection algorithm based on multiscale representation and convolutional neural network

Our algorithm is a group of the following steps as showed in Figure 1. The algorithm requires a pyramidal image representation as a decreasing scale copies set. Scale decreases as we go higher on the pyramid. Images are broken into blocks at each pyramid level. There is object detection in each block with an application CNN. It is necessary to divide the image into minimal number of parts, so that the object on the CNN input block is not fragmented. Therefore, overlapping blocks are used. Moreover, the higher overlap amount, the larger object can be detected without the probability of its division into parts. However, it will require additional computational cost. After image processing using CNN, a procedure for combining the found regions at all levels is required. Thus, our algorithm includes four basic steps.

1. Determination of the number of levels in the image pyramid. It should be borne in mind that the top-level size should be close to the input layer size for used CNN. Number of levels is determined considering possible image rotation as:

$$P = \left[\max(W, H) / l\right] - 1, \tag{1}$$

where W, H the input image width and height; $l \times l$ - the used CNN input layer size; [*] - nearest integer.

This approach eliminates objects fragmentation at a minimum scale and allows them to increase their correct classification as a whole in image.

2. The image is separated into blocks with overlapping on the p layer. Their number is determined by the expression:

$$B_p = \left(\frac{W / p - l}{k} + 1\right) \cdot \left(\frac{H / p - l}{k} + 1\right) = \left[B_{W_p}\right] \cdot \left[B_{H_p}\right], \tag{2}$$

where p is the pyramid level number, $p=1, \ldots, P$; k-block shift amount, in pixels.

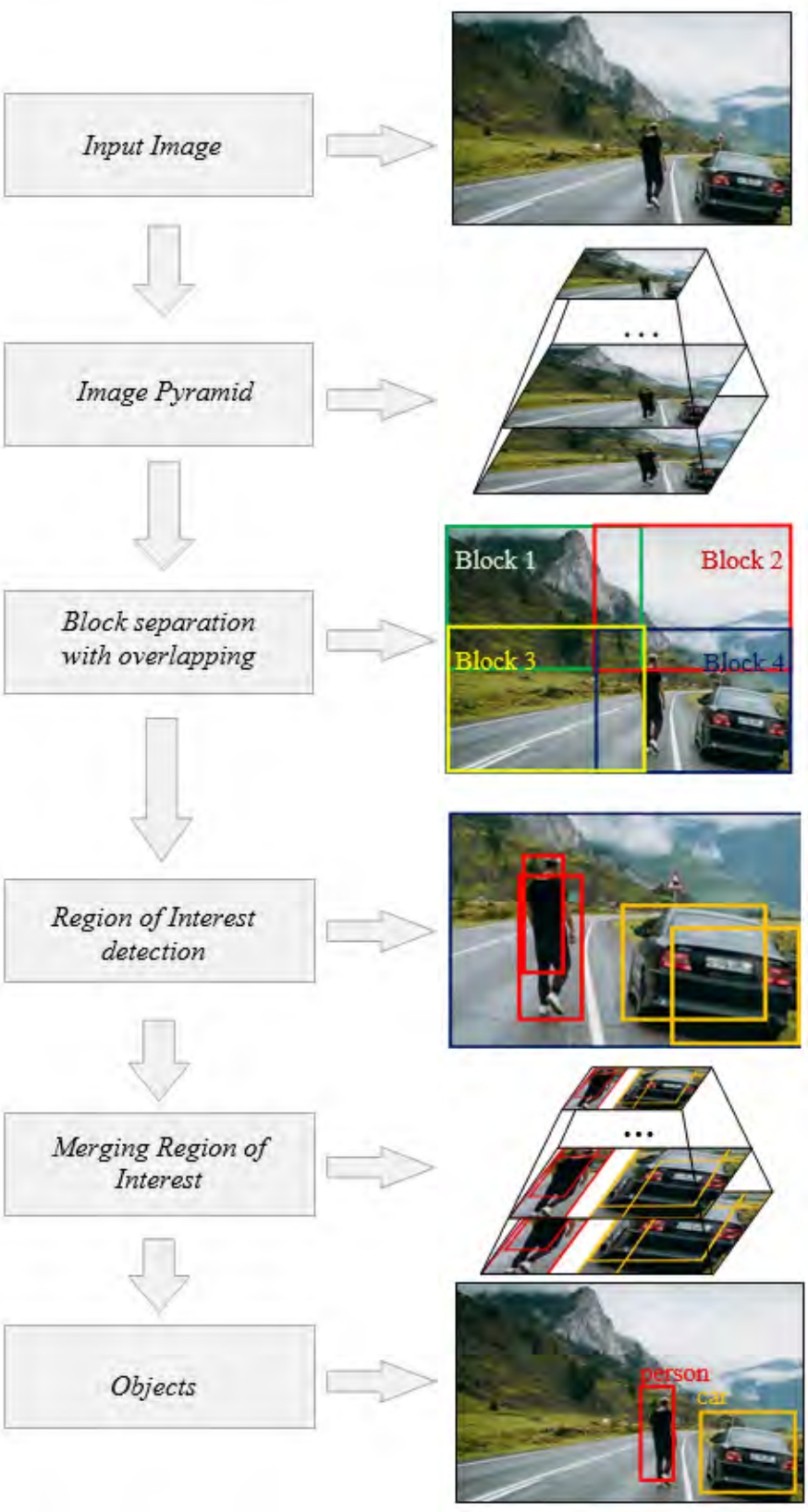


Figure 1: Flow chart of our proposed algorithm

The last incomplete block is merged with the previous block and scaled to the input CNN layer size when the B_{W_P} or B_{H_P} is rounded down. Otherwise, if the block size is rounded up, then the block values are padded with zeros.

The block overlap number can be calculated using the equation:

$$\alpha = \frac{k}{l} \cdot 100\,\%, \tag{3}$$

3. Region of interest (RoI) detection that may contain an object or the object fragment. For this, each block is processed using CNN, and the detected RoI is described by a feature set:

$$F = \left(x_1, y_1, x_2, y_2, E, Cl\right), \tag{4}$$

where x_1, y_1 - upper left corner coordinates of the found region in the image; x_2, y_2 – lower right corner coordinates of the found region in the image; Cl – the selected object class; E is CNN confidence in the correct classification.

4. Merging ROI.

4.1. Localization of coordinates of the object at different levels of pyramid can be determined with slight deviations. In addition, if the object is large, then its fragment can be identified as an independent ROI at some middle or lower levels. Therefore, we use an analysis of the overlaps number at all levels and blocks, as well as belonging to the same class to combine ROI O_i and O_j:

$$IoU(O_i, O_j) > 0{,}5,\ Cl_i = Cl_j, \tag{5}$$

where

$$IoU(O_i, O_j) = \frac{O_i \cap O_j}{O_i \cup O_j}, \quad i, j \in 1, N \mid i \neq j, \tag{6}$$

The merged region $F(O')$ descriptor is formed as follows:

$$F(O') = \begin{pmatrix} \min\left(x_1^{O_i}, x_1^{O_j}\right), \min\left(y_1^{O_i}, y_1^{O_j}\right), \max\left(x_2^{O_i}, x_2^{O_j}\right), \\ \max\left(y_2^{O_i}, y_2^{O_j}\right), \max\left(E^{O_i}, E^{Oj}\right), Cl\left(\max\left(E^{O_i}, E^{Oj}\right)\right) \end{pmatrix}. \tag{7}$$

4.2. The same object or its fragments can be detected with a coordinate shift at different scales or in neighboring cells during block detection results processing. In addition, the smaller size object fragment features may by different from the original object features and even be closer to another class descriptors. Therefore, post-processing is additionally applied when combining process. To do this, it is proposed to use the rule according to which all areas obtained in the previous step are analyzed and combined:

$$IOU(O'_i, O'_j) > 0{,}2,\ Cl_i = Cl_j,\ \ d_x > t_x,\ d_y > t_y, \tag{8}$$

where d_x and d_y are the difference values between the candidate regions sizes along the x and y axes, respectively; t_x and t_y are threshold levels.

The false objects merging probability of the same class that are located at an insignificant distance from each other will be increased if the threshold value is decreased at step 4.1.

4. Experimental results and discussion

The object detection algorithm was tested on a sample of 820 4K images. These images were taken under different weather conditions, with different mounting heights and camera tilt angles. The annotated image dataset was prepared using labelImg tool [13]. We used YOLOv4 implementation from [14]. The annotated images dataset includes 6125 objects of two classes ("person" and "vehicle"). The objects sizes vary from [24×14] to [322×780]. The "vehicle" class is a composite class because it includes the "bus", "car", and "truck" classes. In Figure 2 total marked 11 objects, of which 3 belong to the class "person", and 8 belong to the class "vehicle".

The proposed algorithm is implemented in the Python programming language using the PyTorch machine learning framework for experiments. Basic operations on images are implemented using the OpenCV computer vision library. To increase the detection speed, batch image blocks processing for each pyramid level is applied using CUDA technology.

We calculate mAP (mean Average Precision) metric to assess the performance of the algorithm. Metric mAP is calculated by taking the mean AP over all classes. At the same time, at the first stage, the number of true positive T_p and false positive F_p detections were calculated. For this, intersection over union (IoU) metric was applied for the detected and annotated objects on the images of the dataset with a threshold level $T = 50\,\%$ for making a decision. Then the precision p and recall r were calculated:

$$p = \frac{T_p}{T_p + F_p}, \tag{9}$$

$$r = \frac{T_p}{T_p + F_n}, \tag{10}$$

where F_n - false negative detections

Based on the obtained values, the AP metric was calculated according to the formula (9) for each class for 11 threshold levels in the range from 0 to 1 with a step (0.1) [15]:

$$\mathrm{AP} = \frac{1}{11} \cdot \sum_{i=0}^{10} p_{\mathrm{int}}(r_i) \tag{11}$$

where $p_{\mathrm{int}}(r_i) = \max\, p(\tilde{r})$, $\tilde{r} \geq r_i$.

At the last stage, we calculate the mAP metric obtained in the previous step for two object class.

Figure 2: Examples of annotated images for test

To use the CNN capabilities more optimally when detecting and identifying small size objects, it is necessary to determine the CNN threshold confidence and the input layer size, when the greatest work efficiency. Experiments have shown that the smaller the scaling, the greater the accuracy in detecting small objects in high-resolution images. So, with the input layer size [608 × 608], mAP is 22%, and with the size [1024 × 1024] the highest value for mAP is 48% is achieved on a test images set with 4K resolution, while the confidence threshold is $T = 50\,\%$. We used computer with CPU Intel Core i9-10900 2.8 GHz, 64 GB RAM, NVIDIA GeForce RTX 2080 Ti. Based on step one of our algorithm and more effective layer size 4K resolution image must be presented in the pyramid form of three levels for processing.

At the experiments next stage, the algorithm accuracy was evaluated for various block overlap and CNN threshold value with a step of 5%. It should be noted that for the minimum original image scale,

threshold value of $T = 50\,\%$, which corresponds to the most effective for the used CNN. The experimental results are presented in Table 1. The analysis shows that for any α and T values, the considered algorithm's mAP is higher than the mAP of the CNN used. The best result is provided with $\alpha = 50\,\%$ and $T = 75\,\%$, and mAP = 68.3%.

Figure 3 shows processed 4K resolution parts of images from the prepared dataset. We can see in Figure 3a two detected objects using YOLO v4 and four detected objects using our algorithm in Figure 3b.

Table 1

Influence of overlap and confidence threshold on object detection accuracy, mAP,%

	α			
T	40	50	60	70
65	64.6%	68.0%	62.7%	61.6%
70	66.6%	67.6%	63.5%	62.8%
75	66.3%	**68.3%**	65.8%	64.4%
80	62.0%	62.4%	65.2%	63.7%

a)

b)

Figure 3: There are 4K-image parts with examples of object detection: a) using YOLOv4; b) using the proposed algorithm

Also, the proposed algorithm was tested on 4K [16] and 8K [17] images. Figure 4 shows the object detection result by the proposed algorithm with parameters which were determined as a result of experiments: image 4K with resolution [3840×2160], input layer size [1024×1024], number of pyramid levels P=3, $\alpha = 50\,\%$ and $T = 75\,\%$. In this case, 90 objects were detected, their minimum size is [33×11] pixels. Small objects give the greatest gain in object detection based on presented algorithm for 4K images (Figure 4).

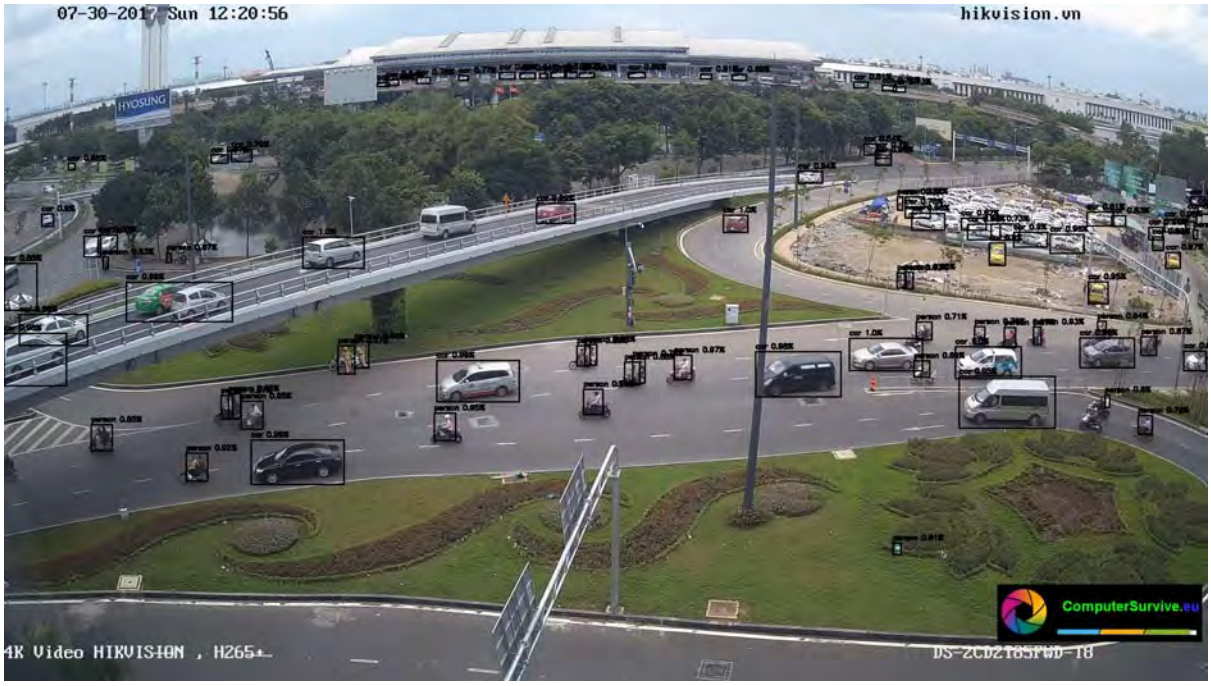
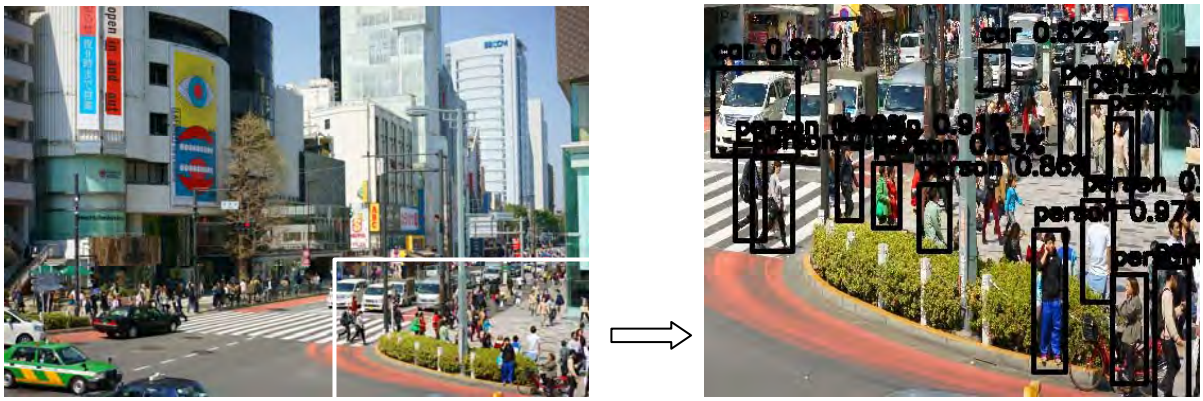


Figure 4: Examples of object detection on a 4K image based on proposed algorithm

Figure 5a shows a reduced copy for 8K image with resolution [7360×4912]. YOLOv4 based image processing result is proposed in Figure 5b. On this image 27 objects were detected. The 8K image has a huge size, so object detection result is presented only for bottom right part of the image. Our algorithm detected 114 objects and their minimum size is [26×31] pixels on this image. Figure 5c shows the object detection result by the proposed algorithm with parameters which were determined as a result of experiments: input layer size [1024×1024], number of pyramid levels P=4, $\alpha = 50\,\%$ and $T = 75\,\%$. Figure 5c shows that small objects give the greatest gain in object detection on image 8K resolution. We can see (Figure 5c) that the presented algorithm even detected the drivers of vehicles, who stopped in front of the pedestrian crossing.

5. Conclusion

This article proposes the algorithm for detecting objects in 4K and 8K images, which includes the following basic steps: forming an image pyramid until the top level dimensions are close to the input layer dimensions of the used CNN; overlapping block splitting for all resulting layers; using CNN for each block object detection; post-processing for the results obtained in the previous step. We used a YOLOv4 CNN for object detection with an input layer size of [1024×1024]. It gives high efficiency in detecting small objects in 4K and 8K resolution images. The proposed algorithm was implemented using Python with PyTorch machine learning framework, computer vision library OpenCV and CUDA technology. To conduct experiments to assess the proposed algorithm effectiveness, a high resolution images dataset with annotated small and medium size of objects of two classes ("person" and "vehicle") was prepared. Experiments have shown that mAP metric can reach 68.3% for the data used. The obtained results indicate that the proposed approach is promising for solving applied problems of small object detecting in high resolution images. In some cases, if objects of one class are located nearby, then possibly our algorithm will mistakenly merge them. Therefore, our further work will be done to solve this problem.



a) b)

c)

Figure 5: Examples of object detection on 8K image: a) reduced copy of 8K image; b) using YOLOv4 algorithm; c) using the proposed algorithm

6. References

[1] S.A. Rajjak, A. K. Kureshi, Recent Advances in Object Detection and Tracking for High Resolution Video: Overview and State-of-the-Art, in: Proceedings of the 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA), Pune, India, 2019, pp.1-9. doi:10.1109/ICCUBEA47591.2019.9128812.

[2] K. Goulekas: Visual Effects in a Digital World: A Comprehensive Glossary of over 7000 Visual Effects Terms, Morgan Kaufmann, San Francisco, CA, 2001.

[3] D. Vorobjov, I. Zakharova, R. Bohush, S. Ablameyko, An effective object detection algorithm for high resolution video by using convolutional neural network, volume 10878 of Lecture Notes in Computer Science, Springer-Verlag, Cham, 2018, pp. 503–510. doi:10.1007/978-3-319-92537-0_58.

[4] L. Yongxi, T. Javidi, Efficient object detection for high resolution images, in: Proceedings of 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 2015, pp. 1091–1098. doi:10.1109/ALLERTON.2015.7447130.
[5] J. Xiao, J. Hays, K. Ehinger, A. Oliva, A. Torralba, Sun database: large-scale scene recognition from abbey to zoo, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'10, San Francisco, CA, USA, 2010, pp. 3485–3492. doi:10.1109/CVPR.2010.5539970.
[6] V. Ruzicka, F. Franchetti, Fast and accurate object detection in high resolution 4K and 8K video using GPUs, in: Proceedings of IEEE High Performance Extreme Computing Conference (HPEC), Waltham, MA, USA, 2018, pp. 1–7. doi:10.1109/HPEC.2018.8547574.
[7] P. Korshunov, T. Ebrahimi, UHD video dataset for evaluation of privacy, in: Proceedings of 6th International Workshop on Quality of Multimedia Experience, QoMEX'14, Singapore, 2014, pp. 232–237. doi:10.1109/QoMEX.2014.6982324.
[8] F.O. Unel, The power of tiling for small object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW 2019), Long Beach, CA, 2019. Doi: 10.1109/CVPRW.2019.00084.
[9] J. Redmon, S. K. Divvala, R. B. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'16, Las Vegas, NV, USA, 2016, pp.779–788. doi:10.1109/CVPR.2016.91.
[10] J. Redmon, A. Farhadi, YOLO9000: Better, Faster, Stronger, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE Press, Washington DC, 2017, pp. 6517–6525. doi:10.1109/CVPR.2017.690.
[11] J. Redmon, A. Farhadi, YOLOv3: An Incremental, 2018. URL: https://arxiv.org/abs/1804.02767.
[12] A. Bochkovskiy, Ch.-Y. Wang, H.-Y. M. Liao, YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020. URL: https://arxiv.org/abs/2004.10934.
[13] LabelImg is a graphical image annotation tool and label object bounding boxes in images, 2018. URL: https://github.com/tzutalin/labelImg.
[14] Pytorch-YOLOv4. URL: https://github.com/Tianxiaomo/pytorch-YOLOv4.
[15] Everingham M., Van Gool L., Williams C., Winn J., Zisserman A. The pascal Visual Object Classes (VOC) challenge. International Journal of Computer Vision 88, 2010, pp. 303–338. doi:10.1007/s11263-009-0275-4.
[16] HIKVISION DS-2CD2T85FWD-I8 8MP camera H265+ 4K, 2020. URL: https://www.youtube.com/watch?v=nG_wGUi-Ozc.
[17] Tomo.yun, 2021. URL: http://www.yunphoto.net/he/photobase/hr/hr22577.html.