Journal on Internet of Things
DOI:10.32604/jiot.2020.09814
Article

A Real Time Vision-Based Smoking Detection Framework on Edge

Ruilong Chen[1], Guangfu Zeng[1], Ke Wang[2], Lei Luo[1,*] and Zhiping Cai[1]

[1]College of Computer, National University of Defense Technology, Changsha, 410073, China
[2]College of Computer, Guizhou University of Finance and Economics, Guiyang, 550025, China
[*]Corresponding Author: Lei Luo. Email: l.luo@nudt.edu.cn
Received: 20 January 2020; Accepted: 05 June 2020

Abstract: Smoking is the main reason for fire disaster and pollution in petrol station, construction site and warehouse. Existing solutions based on wearable devices and smoking sensors were costly and hard to obtain evidence of smoking in unmanned scenarios. With the developments of closed circuit television (CCTV) system, vision-based methods for object detection, mostly driven by deep learning techniques, were introduced recently. However, the massive GPU computing hardware required by the deep learning algorithm made these methods hard to be deployed. This paper aims at solving the smoking detection problem on edge and proposes the solution that has fast detection speed, high accuracy on micro-objects and low computing budget, i.e., it could be deployed on the edge device such as NVIDIA JETSON TX2. We designed a new framework named RTVBS based on yolov3 and made a smoking dataset to train our model. We raised several methods to improve detection accuracy during the training step. The validation results show our model has excellent performance in smoking detection.

Keywords: Smoking detection; small object detection; real time; CNN; image processing

1 Introduction

Nowadays, safety and health are still the focus of public attention, as one of the most dangerous daily behavioral habits, smoking has caused plenty of fire hazards and other kinds of catastrophes. According to the investigation, there are around 5.2% of the total 312,000 fire events caused by smoking every year, causing thousands of casualties. Besides, World Health Organization attributes 10% annual early deaths to cigarettes and proves it increased risk for many serious diseases. In addition, smoking is also a main factor leading to air pollution. Therefore, preventing smoking is of great significance.

It is not difficult to detect smoking behaviors indoors, such as malls, colleges, restaurants. We can deploy smoking sensors and CCTV systems for detection. These indoor places were located near the data centers with ultrahigh bandwidth. It has the abilities to process requests and video data timely through the central server. However, in edge scenarios [1], the hardware resources are limited [2], such as weak bandwidth and network, we could not connect to central server timely. Besides, the sensor-based methods for detection do not work well outdoors. For example, smoke sensors find out the smoking events relying on the fumes concentration, but in outdoor environments, the fumes concentration is greatly diluted, this badly disturbs the detection results. A light-weight vision-based smoking detection methods might be the best choice on edge [3].

Since AlexNet won the first prize on IMAGENET classification contest in 2012 [4], researchers gradually paid more and more attention to deep learning based methods on computer vision. Recently, object detection tasks utilized the CNN algorithms to solve practical problems successfully, such as face

detection, pedestrian detection and vehicle detection. The results are quite satisfying. There is too little work has been devoted to smoking detection by means of CNN. Therefore, so in this paper, we apply deep learning methods to perform smoking detection.

Taking detection speed into consideration, we chose the one-stage detection algorithms for their high speed. Two-stage detection algorithms have high precision and perform well in small object detection, but their model parameters are very large and have a long period of detection which make they are difficult to satisfy the need for real-time processing.

Though CNN has an advantage in extracting features from objects, cigarette's features are not obvious, many little things might affect the detection outcomes. So, we use shapes of hand and mouth as features relevant to smoking events. Then, we build a smoking dataset to train our network and valid the detection results. We referred to the yolov3's network architecture and proposed the RTVBS framework which was compatible for embedded board and it achieved significantly higher accuracy than tiny-yolov3 with almost the same computing speed. Finally, we built a detection system with the trained RTVBS models. The rest of this paper is organized as follows: Section 2 provides the review of the related work. Section 3 introduces the dataset we have built. Section 4 describes the new model we proposed and some improvements as well as experiments on smoking detection. Finally, in Section 5, we present conclusion and future work.

2 Related Work

There were many researches on smoking detection, mostly they used wearable devices [5] and smart sensor to sense the movements of someone's arm or the fumes concentration to detect smoking events [6]. But in the stations, it is impossible to equip everyone including passengers with smart sensors. Zheng et al. [7] utilize the changing of WIFI signal to identify the smoking activities, but as the author said, it works well only indoors. In outdoor environments, this approach is not so effective. Wu et al. [8] used traditional vision-based methods for detection, firstly, they extracted the region of interested objects from the background, secondly, they used color's changes in chromatic ratio histograms of objects to retrieving actions and recorded the change sequences. Thirdly, they used Markov models of different event types to discriminate sequence records, then to figure out smoking events. This method didn't have fast speed and it was easily disturbed by some actions like drinking and eating.

Traditional vision-based object detection techniques were mainly depended on handcrafted features. The lack of effective image representation forced the researchers to design sophisticated feature representations. Moreover, these methods' generalization ability and anti-interference ability were quite poor, around 2010, these methods reached a bottleneck. Since 2012, the world saw the rise of CNN, it has extra ordinary capability to learn robust and high-level feature representations. Soon after that, experts successfully applied the CNN to solve object detection difficulties. The following are milestones in object detection history.

2.1 Residual Learning Framework

Though the Resnet [9] was designed for classification tasks instead of detection tasks, the deep thoughts behind the network left the scholars valuable illumination. It used the residual module to training deep CNN as pioneer and successfully solved the degradation and vanishing/exploding gradients problem what occurred when the amount of network layers increasing. Residual module learns mapping functions with reference to the layer inputs. The author proved the residual networks were easier to be optimized and gained higher accuracy from great amount of network layers, and converged much faster than the ordinary deep CNN structures.

2.2 The anchor from Faster RCNN

In 2015, the first end-to-end and nearly real-time deep learning object detector, Faster RCNN, was invented. It proposed novel Region Proposal Networks that shared convolutional neural network layers with fast RCNN's backbone [10]. RPN were designed to generating detection region proposals with multiple scales and aspect ratios. Then the author introduced translation-invariant box, anchor, which acted as regression references for sharing features without extra cost for addressing multiple scales and aspect ratios. Besides, usage of anchors also reduced model's elapsed time.

2.3 Feature Pyramid Networks

The FPN architecture [11] was proposed on basis of Faster RCNN in 2017, it was a top-down architecture with lateral connections which was developed for building high-level semantic feature maps at all scales. The contribution of FPN was constructing feature pyramids with little marginal extra cost. The feature pyramids were formed naturally through CNN's forward propagation. The FPN showed significant improvement as a generic feature extractor for detecting objects with different scales by integrating multiscale feature representations from each level of feature pyramids. Furthermore, low-level features had small receptive field but higher-resolution feature maps what is in favor of detecting small objects.

2.4 YOLO Family

As a leader of one-stage detectors in 2016, YOLO algorithm gave the object detection task a totally new definition, it treated detection problem as a regression problem and utilized a single neural network to predict bounding boxes and class probabilities directly from a raw input image in one evaluation [12]. Since the single network architecture, the inference speed of YOLO algorithm is extremely fast. To improve the detection accuracy while keeping a very high detection speed, YOLO's family members, yolo9000, yolov3 and tiny-YOLO were proposed one after another. The improvements of all these different network architectures offer us enlightenment to design our RTVBS.

3 Model Design

Our vision-based method for smoking detection used deep learning network to located the smoking objects from a single picture, in other words, we treat the smoking detection problem as an object detection problem. The objects we want to detect are cigars. There were a lightweight and fast detection framework called tiny-yolov3, it could perform detection with little time cost, but its accuracy is not high enough and has a poor performance in detecting micro objects such as cigars, hands and mouths. Generally, deeper neural network learns more characteristics and features of targets. The architecture of tiny-yolov3 is excessively simple, we should design a more sophisticated network structure to improve accuracy. Besides, we should place restrictions on the amounts of model's parameters and cut down the model's hardware resource demands to make it adapt for the embedded system environment. More importantly, the new model should detect tiny objects more accurately.

The following were descriptions about the detection procedures. First, we trained an improved detection model with the training dataset we built. Then we deployed the model to edge embedded board. The board was connected to several surveillance cameras via local area network and the board processed the RSTP video stream from those cameras. When smoking events occurred, the board would capture the video frame and mark the cigar's location, and then send alert to the central console which was also in the LAN and managed by security guards. It is obvious that the model's performance is the most important part in our framework.

3.1 Architecture

Fig. 1 demonstrates the whole architecture of our RTVBS. In general, invariance and equivariance are two important properties in one image's feature representations. The feature maps near inputs have higher resolution and small receptive fields that help to discover small scale objects but hinder learning high-level semantic information for classification, we think these feature maps have weaker invariance

but stronger equivariance. It is essential to balance the invariance and equivariance problems, so that we can detect objects with different scales more accurately. Inspired by the FPN and Yolov3 which use the feature fusion strategy, we add a larger scale feature map's information into the original predictions with skip connections on the basic structure of YOLO, as Predict 2 and Predict 3 in Fig. 1 shows. Finally, we combine the three levels of predictions to figure out the final detection. This is helpful for finding out small objects. HyperNet [13] adopts the approach concatenating the different features of multiple layers, nevertheless, the feature maps of different layers may have various spatial and channel dimensions, we settle it with up-sampling operation.

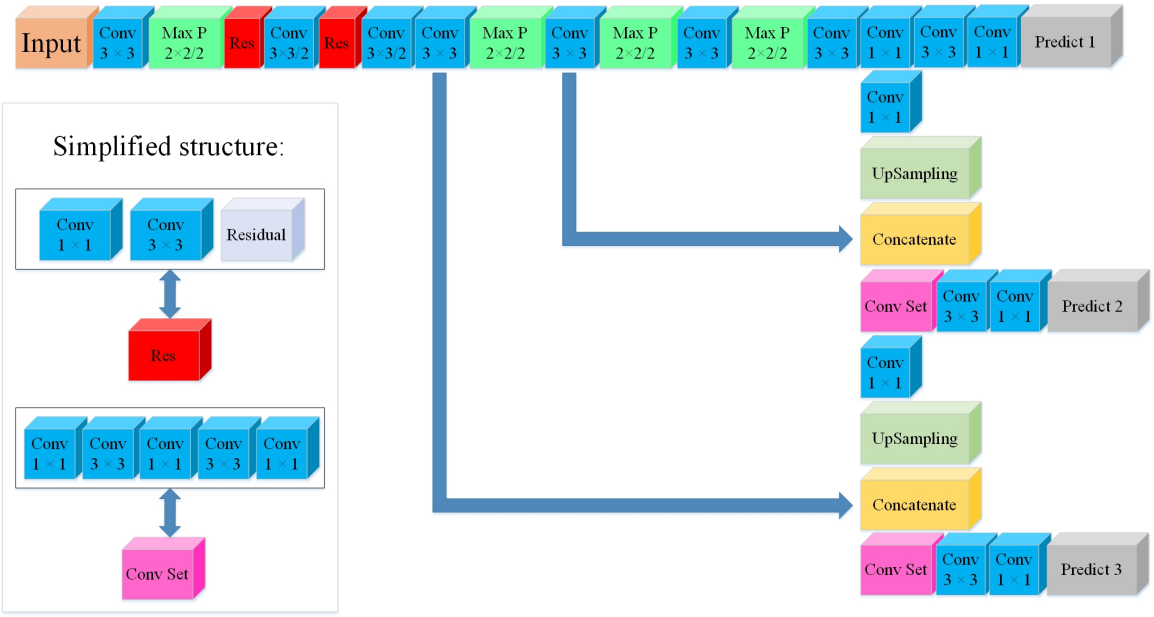

Figure 1: The total 44 layers architecture of RTVBS

As mentioned in Section 2.1, residual module has advantages in the aspects, it helps model learning more features and converged faster than ordinary convolutional neural network. We chose to add two residual modules at our model's backbone in order to make RTVBS learning more cigar's features so that it can detect more accurately. The details of residual module are also in Fig. 1.

The ConvSet in Fig. 1 contains five convolutional neural network layers with different convolution kernels. The design of its architecture is inspired by the GoogLeNet's Inception [14]. Consequently, the amount of our model's parameters can be reduced, thus, our framework's calculation speed is improved further more. It also helps to balance invariance and equivariance of the model and learning more comprehensive feature representations.

3.2 Improvements for Training

We adopt two strategies in the training steps.

3.2.1 Random Training Shape

According to the paper [15], random shape inputs reduce the risk of overfitting in model training and make model more robust in different detection situations. When training a batch of images, we resize them to an equal size, the size of different batches is not equal. In our RTVBS, there are total 6 times of down-sampling, including max pooling and convolution operation with two strides in one step. The final feature map has shrunk 64 times. Thus, the width and height of inputs should be a multiple of 64 and We calculate them as Eq. (1), Referring to Over Feat [16], we set convolutional neural network layers instead of fully connected layers to perform predictions. In this way, our model is able to handle random shape inputs.

$$\begin{cases} L_{width} = L_{height} \\ L_{height} = 320 + 64k, k \in \{0,1,2,3,4,5\} \end{cases} \tag{1}$$

3.2.2 K-means for Anchors

Our RTVBS sets up 9 anchors introduced in Section 2.2 in 3 prediction layers, it means there are three anchors in one prediction layer. We apply *k*-means clustering algorithm to work out the nine cluster centers according to ground truth boxes' edge lengths in our training dataset. There are three levels of prediction scale, in Fig. 1, feature map in Prediction 3 has the largest size (width × height), feature map of Prediction 2 takes the second place and Prediction 1's feature map is smallest. We sort the size of the calculated anchors (width × height) and select the three smallest anchors to the Prediction 3 for its largest feature map scale, these feature maps have small receptive field, so it can find out and locate small objects. Then, the three largest anchors to Prediction 1 and the left anchors for Prediction 2. The coverage of multiple scales of targets is guaranteed in this way. By using anchors, our model can finish bounding box regression procedure in a shorter time and locate objects more precisely.

4.1 Training methodology

Yolov3 tried the focal loss proposed by RetinaNet [17] but did not work well, so we used binary cross-entropy loss for the class predictions rather than soft max. As for bounding box regression loss, we used the Mean Squared Error to measure it. Our RTVBS's loss function as indicated below:

$$\begin{aligned} Loss &= \lambda_{coord} \sum_{i=0}^{S\times S} \sum_{j=0}^{B} \mathrm{I}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] + \lambda_{coord} \sum_{i=0}^{S\times S} \sum_{j=0}^{B} \mathrm{I}_{ij}^{obj} (2 - w_i \times h_i)[(w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2] \\ &- \sum_{i=0}^{S\times S} \sum_{j=0}^{B} \mathrm{I}_{ij}^{obj} [\hat{C}_i \log(C_i) + (1 - \hat{C}_i) \log(1 - C_i)] - \lambda_{noobj} \sum_{i=0}^{S\times S} \sum_{j=0}^{B} \mathrm{I}_{ij}^{noobj} [\hat{C}_i \log(C_i) + (1 - \hat{C}_i) \log(1 - C_i)] \\ &- \sum_{i=0}^{S\times S} \sum_{j=0}^{B} \mathrm{I}_{ij}^{obj} \sum_{c \in classes} [\hat{p}_i(c) \log(p_i(c)) + (1 - \hat{p}_i(c)) \log(1 - p_i(c))] \end{aligned} \tag{2}$$

Definitions of parameters in the function are same with those in YOLO's loss function. λ_{coord} and λ_{noobj} are two weight hyper-parameters for changing part of final loss. S means our system divides the input image into an S × S grid and for each grid cell predicts B bounding boxes. S is equal to the feature map's width of last CNN layer. In our models, B's value is three, which is same as the amount of anchors in one prediction level. I_{ij}^{obj} denotes that the jth bounding box predictor in cell i is "responsible" for that prediction. I_{i}^{obj} denotes if object appears in cell i. I_{ij}^{noobj} is opposite to the I_{ij}^{obj}. C means the number of labeled categories. The (x_i, y_i) represent the center coordinate of the box relative to the bounds of the gridi. The (w_i, h_i) are width and height relative to the whole image's width and height. The $\hat{\mathrm{p}}(\mathrm{c})$ means the probability of the predicted class. The $\hat{\mathrm{x}}, \hat{\mathrm{y}}, \hat{\mathrm{w}}, \hat{\mathrm{h}}, \hat{\mathrm{C}}, \hat{\mathrm{p}}(\mathrm{c})$ are the ground truth values from the training dataset and the x, y, w, h, C, p(c)are the predicted value from the model.

We borrowed the batch normalization [18] to address the internal covariate shift problem and to accelerate the back-propagation between the CNN layers. Besides random input shapes, we adopt other data augmentation techniques applied in Yolov3 such as random cropping, color jittering and flipping. Finally, in the prediction layers before the final outputs, we use non maximum suppression [19] methods to delete duplicated bounding boxes.

We performed 50 epochs of training with ainitial learning rate of 0.001 and divided it by 10 at 40 and 45 epochs. The weight decay was set to 0.0005 and moment umparameter was set to 0.9. Our RTVBS was implemented in the Darknet framework for training.

4 Experiment

Primarily, we created a dataset which was relevant to the smoking behaviors. Then we utilize the dataset to train our RTVBS model. Finally, we did several experiments with the dataset, after that we analyzed the experimental results.

4.1 Build Dataset

In the era of deep learning, data is the most essential elements to establish a high-performance model. The quality of model depends on the quality of the training data. But there is no smoking behavior dataset available on the Internet as far as I know, so, we created the smoking behavior dataset which was appropriate for the practical detection situation. The dataset contained three parts, the pictures from the Internet, the frame from smoking video we recorded and the frame from movies and HMDB we captured.

4.1.1 Introduction of the Dataset

The total number of pictures for training is about 2000, including the 1200 screenshot pictures captured from the online videos and HIKVISION surveillance cameras. And the remained 800 images were crawled from some photo websites. The features extracted by CNN mainly include contour, color, chromatic aberration, shape, size, texture and the combination information of these basic graphics features. If we only mark the box around the cigar as ground truth, there will be two flaws. The first one is that the cigar is too small to be labeled, we may make a wrong label. The second is due to occlusion, blur and other interference factors, the cigar's graphic characteristics and features are not obvious, our model may learn incorrect features, which is harmful for detection. Considering cigar is held on the hand or stuck between lips, we regard hands and mouths as features of cigar objects.

(a) Smoke_Hand_Lip (b) Smoke_On_Hand (c) Smoke_With_Lip

Figure 2: Samples of smoking category

(d) distractor one (e) distractor two (f) distractor three

Figure 3: Samples of smoking distractor

According to the specific patterns of normal smoking steps, there are three smoking status. Fig. 2 presents three categories of smoking. To enrich the dataset, diversify the data and consequently enhance

the accuracy and robustness of our detection model, disturbing samples are necessary. Fig. 3 presents three kinds of distractor and Tab. 1 gives the explanations.

The sizes of cigars are various in our dataset, which is beneficial for enhancing robustness and generalization of our detection model. Certain pictures contain multiple smokers, which is in favor of improving the multi-target detection results. Some pictures contain the smokers who are far away from the cameras, which is of great benefits to improving the results of small target detection. The diversified background, including indoor, outdoor, street and natural scenes, also improves the robustness of model.

Table 1: Explanation of smoking categories

Name	Explanation	Position
Smoke Hand Lip	Smoker holds the cigar near the lips with hand	Fig. 2a
Smoke on Hand	Smoker just holds the cigar in the hand far from mouth	Fig. 2b
Smoke with Lip	Smoker just sucks the cigar between lips without hand	Fig. 2c
Distractor one	As the interference term of Fig. 2a	Fig. 3a
Distractor two	As the interference term of Fig. 2b	Fig. 3b
Distractor three	As the interference term of Fig. 2c	Fig. 3c

4.1.2 Data Annotation

We chose the open source annotation tool YOLO Mark to label our data. The contents of annotation were composed of object's category and ground truth box. For every picture in the dataset, we created a text document which duplicated the picture's prefix name to store the label messages. There might be multiple detection objects in one picture, so the text file might contain several lines, one line denoted an object, its format was quintuple: (class, x, y, w, h). Class meant the category of the object. The x, y indicated the horizontal and vertical coordinates of the labeled box's center point, the w, h represented the width and height of the box respectively. For disturbing samples, we didn't give them annotation and just created empty text files.

4.1.3 Composition of Dataset

We divided the labeled dataset into training set and test set. Tab. 2 gives the detailed information about the composition of different classes in the dataset.

Table 2: Statistics of dataset

	Name of class	Quantity		Name of class	Quantity
Training Set	Smoke Hand Lip	600	Test Set	Smoke Hand Lip	60
	Smoke on Hand	600		Smoke on Hand	60
	Smoke with Lip	600		Smoke with Lip	60

4.2Comparison Experiment

We chose the RetinaNet, Yolov3 and Tiny-yolov3, three high performance frameworks in one-stage detection algorithms to make a comparison for our RTVBS. Tab. 3 describes the detailed experiment results of four trained model. By contrast, we gave up the RetinaNet for its slow detection speed and Yolov3 for its large model size and parameter quantity. Tiny-yolov3 seems to be the most suitable model for our detection scenarios, so we designed RTVBS on the basis of it. Finally, we train and valid the RTVBS with the same training dataset, the results are also contained in Tab. 3.

4.3 Results and Analysis

We deployed the four trained models into the NVIDIA JETSON TX2 and test their performance with the test dataset we created before. Tab. 3 describes the detailed test results, additionally, Fig. 4 demonstrates the outcomes more clearly.

Table 3: Descriptions of the experiment results

	RetinaNet	Yolov3	Tiny-Yolov3	RTVBS
Language	Python	C++	C++	C++
Framework	Keras-Tensor flow	Darknet	Darknet	Darknet
Model size (MB)	146.27	246.35	34.72	36.93
Detection rate (FPS)	1.32	3.72	14.42	13.63
mAP (%)	72.32	69.18	58.57	69.44

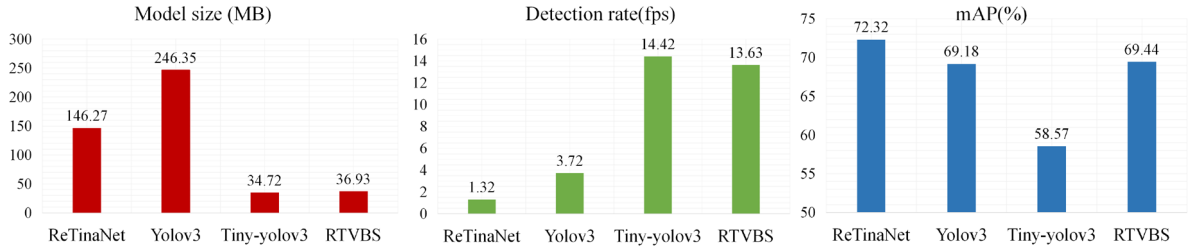

Figure 4: Histograms of four models' comparisons

We picked three benchmarks to measure the models' performance. Model size is usually related to the number of parameters and reflects the computing demands. Larger model contains more parameters and it needs stronger GPU processor and more GPU memory. Our RTVBS is a lightweight model which consumes less hardware resources. The detection rate directly reflects whether a model is a real-time detection model, our RTVBS is up to 13.6 fps, which makes it suitable for real-time detection. The last benchmark is mAP, it was proposed in PASCAL VOC dateset and reflected the detection accuracy of a model. From Fig. 4, we know that the RetinaNet and Yolov3 have high detection accuracy but the low detection speed. Tiny-Yolov3 has the fastest detection speed, simultaneously, it has the worst detection accuracy. In my view, we only detected three types of objects, the CNN layers of model need not be too much, a well-designed lightweight model could solve the smoking detection problems. Even if RetinaNet and Yolov3 have good performance in detection, they are not suitable for the embedded board environment where hardware resources are limited.

Taking the three benchmarks into account, only our RTVBS has both relative high accuracy and detection speed.

5 Conclusion and Future Work

In this paper, we designed a high speed and accuracy detection framework which consumed less hardware resources. Besides, we created a smoking behavior dataset to train our RTVBS framework. Its lightweight characteristics promised the widely application on edge. Additionally, we built a detection system on the basis of the trained RTVBS model. And then, we deployed the system into the NVIDIA JETSON TX2 board to performed the detection work. The integration of deep and shallow features in CNN model helps improve both invariance and equivariance works, so our model has good performance in small object detection as shown in Fig. 5.

Our future work will focus on model compression and acceleration, we are considering the model pruning, Huffman coding and other methods. Besides, there are false detection and miss detection cases in

our model, we will expand the dataset and refine the model further more. In addition, we will attempt to deploy the model to the NVIDIA JETSON Nano for practical use.

Figure 5: Detection results on three types of smoking

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] F. Liu, G. M. Tang, Y. H. Z. Li and Z. P. Cai, "A Survey on Edge Computing Systems and Tools," in *Proc. of the IEEE*, vol. 107, no. 8, pp. 1537−1562, 2019.

[2] F. Liu, Y. T. Cuo, Z. P. Cai, Xiao, N., Zhao, Z. M. *et al.*, "Edge-enabled disaster rescue: a case study of searching for missing people," *ACM Transactions on Intelligent Systems and Technology*, vol. 11, no. 4, pp. 1−26, 2020.

[3] Y. T. Guo, F. Liu, Z. P. Cai, N. Xiao and Z. M. Zhao, "Edge-based efficient search over encrypted data mobile cloud storage," *Sensors*, vol. 18, no. 4, pp. 1189, 2018.

[4] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Int. Conf. on Neural Information Processing Systems*, pp. 1097−1105, 2012.

[5] V. Senyurek, M. Imtiaz, P. Belsare, S. Tiffany and E. Sazonov, "Cigarette smoking detection with an inertial sensor and a smart lighter," *Sensors*, vol. 19, no. 3, 2019.

[6] V. Senyurek, M. Imtiaz, P. Belsare, S. Tiffany and E. Sazonov, "Smoking detection based on regularity analysis of hand to mouth gestures," *Biomedical Signal Processing and Control*, vol. 51, pp. 106−112, 2019.

[7] X. L. Zheng, J. L. Wang, L. F. Shangguan, Z. M. Zhou and Y. H. Liu, "Design and implementation of a CSI-based ubiquitous smoking detection system," *IEEE/ACM Transactions on Networking*, vol. 25, no. 6, pp. 3781−3793, 2017.

[8] P. Wu, J. W. Heish, J. C. Cheng, S. C. Cheng and S. Y. Tseng, "Human smoking event detection using visual interaction clues," in *20th Int. Con. on Pattern Recognition*, pp. 1056−1060, 2010.

[9] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 770−778, 2016.

[10] S. Ren, K. He, R. Girshick and J. Sun, "Faster RCNN: towards real-time object detection with region proposal

networks," in *Advances in Neural Information Processing Systems*, pp. 91−99, 2015.

[11] T. Y. Lin, P. Dollar, R. B. Girshick, K. He, B. Hariharan and S. J. Belongie, "Feature pyramid networks for object detection." in *Proc. of the IEEE Conf. on CVPR*, pp. 2117−2125, 2017.

[12] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unifified, real-time object detection," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 779−788, 2016.

[13] T. Kong, A. Yao, Y. Chen and F. Sun, "Hypernet: Towards accurate region proposal generation and joint object detection," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 845−853, 2016.

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 2818−2826, 2016.

[15] K. He, X. Zhang, S. Ren and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *European Conference on Computer Vision*, Springer, pp. 346−361, 2014.

[16] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus *et al.*, "Overfeat: integrated recognition, localization and detection using convolutional networks," arXiv preprint arXiv:1312.6229, 2013.

[17] T. Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 2980−2988, 2017.

[18] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.03167, 2015.

[19] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. of the 2001 IEEE Computer Society Conf. on CVPR*, vol. 1, pp. I, 2001.