IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 24, 2019, accepted July 24, 2019, date of publication July 30, 2019, date of current version August 15, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2931915

Auto-Zooming CNN-Based Framework for Real-Time Pedestrian Detection in Outdoor Surveillance Videos

SAGHIR ALFASLY[1], (Student Member, IEEE), BEIBEI LIU[1,2], (Member, IEEE), YONGJIAN HU[1,2], (Senior Member, IEEE), YUFEI WANG[2], AND CHANG-TSUN LI[3,4], (Senior Member, IEEE)

[1]School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510640, China
[2]Sino-Singapore Joint Research Institute, Guangzhou 510700, China
[3]School of Information Technology, Deakin University, Geelong, VIC 3216, Australia
[4]Department of Computer Science, University of Warwick, Coventry CV4 7AL, U.K.

Corresponding author: Beibei Liu (eebbliu@scut.edu.cn)

This work was supported in part by the Sino-Singapore Joint Research Institute under Grant 206-A018001 and Grant 206-A017023, in part by the Science and Technology Foundation of Guangzhou Huangpu Development District under Grant 2017GH22, in part by the Science and Technology Foundation of Guangdong Province under Grant 2017A050501002 and Grant 2017A030310320, and in part by the EU Horizon 2020 project entitled Computer Vision Enabled Multimedia Forensics and People Identification under Grant 690907.

ABSTRACT One of the challenges faced by surveillance video analysis is to detect objects from the frames. For outdoor surveillance, detection of small object like pedestrian is of particular interest. This paper proposes a fast, lightweight, and auto-zooming-based framework for small pedestrian detection. An attentive virtual auto-zooming scheme is proposed to adaptively zoom-in the input frame by splitting it into non-overlapped tiles and pay attention to the only important tiles. Without sacrificing detection performance, we have obtained a fully convolutional pedestrian detection model which can be run on low computational resources. It has been trained on an outdoor surveillance dataset and evaluated on two specially prepared testing sets of small (far) pedestrians in outdoor surveillance. We have compared our framework performance with different single-step customized pedestrian detectors as well as the two-step detector faster R-CNN. The results validate the efficiency of our framework.

INDEX TERMS Deep convolutional neural network, outdoor surveillance, real-time pedestrian detection, small pedestrian objects, virtual auto-zooming.

I. INTRODUCTION

Pedestrian detection is a significant task in computer vision field. In most modern artificial intelligence applications such as robotics, video surveillance and autonomous driving, pedestrian detection is an important preliminary step for further processing. During the past few decades, although different algorithms were proposed for high efficient pedestrian detection, it remains a challenging task to detect targets with different appearances, body poses, clothing colors, sizes, fully/semi occlusions, or under different camera views, in complex backgrounds. Recently, convolutional neural networks (CNNs) that generate high quality features have achieved great success in different applications of image analysis like image classification, object detection, segmentation, and scene parsing. As a result, different pedestrian detection systems have been developed with CNN features. In the computational hardware industry, a lot of system on chip (SoC) models have been manufactured. In particular, GPU-based SoCs like Drive PX and TX2 have been used to facilitate a variety of applications where deep learning algorithms and high computational loads are needed. However, there is a contradiction between efficiency and quality in that speed is usually in inverse ratio to accuracy. To build a deep-learning-based algorithm and run it on low computational resources is thus a big challenge. This article proposes a new technique to deal with the above problem. The major contributions can be summarized as follows.

1) Introducing an attentive virtual auto-zooming technique for far pedestrian detection in surveillance cameras that doesn't require physical movement by lens. The input frame is split into a defined number of tiles

The associate editor coordinating the review of this manuscript and approving it for publication was An-An Liu.

and processed with batch inferencing. With a tile selection step, only the important tiles of the frame were used for further processing.

2) Introducing a lightweight end-to-end fully convolutional pedestrian detector that can be run on surveillance systems in real-time.

Nowadays, different types of surveillance systems are produced with different types of surveillance cameras such as bullet cameras, dome cameras, and pan–tilt–zoom (PTZ) cameras. One application of the PTZ cameras in surveillance is to detect suspicious events based on the pixel variations on frame sequence, the camera swivels horizontally and tilts up/down vertically then the camera lens is physically adjusted for the zooming purpose. Although PTZ cameras are used for different scenarios, but for most places in outdoor scenarios the bullet surveillance cameras are used. To better detect the pedestrians in the frames captured by these cameras we propose an attentive virtual auto-zooming scheme without physical movement by lens. The proposed technique optimizes the best scene settings on outdoor cameras. After running on the first few frames, this technique puts the input frame into best form for the proposed CNN-based pedestrian detector. The proposed framework automatically controls the zooming-in and detection tasks in real-time on GPU-based SoCs. An output frame example of the proposed framework is shown in Fig. 1. The close-up is added manually to show the details in the selected area, such as the bounding boxes and their corresponding scores.

FIGURE 1. An example of the detection result of the proposed framework. The manually added circle demonstrates the detection performance in more details.

The rest of the paper is structured as follows. In Sect. II, a brief survey of related work is given. In Sect. III, our attentive auto-zooming detection and lightweight CNN-based pedestrian detection model are introduced in detail. In Sect. IV, a brief overview of the utilized outdoor surveillance pedestrian dataset is given and the training process is described in detail. We demonstrate the performance of the proposed framework in Sect. V.

II. RELATED WORK

CNN features are being used widely for object detection. Girshick *et al.* [1] studied the effects of CNN features generalization for object detection, showing a high detection performance by combining multiple low-level image features with high-level context. The authors used different submodels, one for generating region proposals, the other for extracting CNN features, and the last one consisting of a set of SVM classifiers. That algorithm was improved by Girshick [2] by sharing the generated CNN features for region proposals classification. Further improvement was made in [3] by sharing CNN features with both Region Proposal Network (RPN) and detection network. Later on, a large number of deep CNN-features-based algorithms [4]–[8] have been proposed for object detection, leading to greater advances in both detection and localization. Rather than using region proposals, the single-shot detector YOLO [8] predicts bounding boxes and scores directly from the last feature maps. In a different manner, SSD [7] uses multi-scale convolutional predictors on top of several feature maps, resulting in a large number of predictions with different sizes and aspect ratios, which in turn improve the detection performance.

In [12], Domonkos *et al.* proposed a CNN classifier on each frame region, which is defined by the absolute differences of each two consecutive frames. Similarly, in [11], a CNN classifier is used to refine the proposals generated by traditional pedestrian detectors. Ouyang *et al.* used a different strategy in [9] by applying a CNN to extract pedestrian features of the body parts. Then another filters are applied to predict scores for each part. Finally, all parts scores are jointly used to specify whether the processed window encloses pedestrian or not. In [10], an input image is fed into CNN, producing the CNN features which are shared by a group of CNN pedestrian detectors with different sizes. Training a scene-specific pedestrian detector for video surveillance is addressed by Xiaogang *et al.* [13]. They trained a pedestrian detector with positive and negative pedestrian samples, which are detected from input frames. Du *et al.* [25] used semantic pixel segmentation. Simultaneously with another sub-system, they used the SSD [7] detector to generate a large number of predictions which are further refined with a classification procedure.

Many pedestrian detectors employ a region proposal network RPN [3] as a main part of the detector to improve the detection accuracy. In [23], another object detection model was used for pedestrian detection, which consists of three networks. A VGG16 feature extractor [24] is followed by five convolutional layers. The second network is the proposals generator from multi-scale feature maps. The third part which they call the object detection network is used to classify the generated proposals. The above model is similar to Faster R-CNN [3] but with several feature maps. However, this additional complexity makes the model infeasible for real-time applications. In [26], Zhang *et al.* used a region proposal network [3] to generate candidates before classifying with a Cascaded Boosted Forests algorithm. Brazil *et al.* [29] borrowed the ideas from RPN+BF [26], MS-CNN [23] and [25]

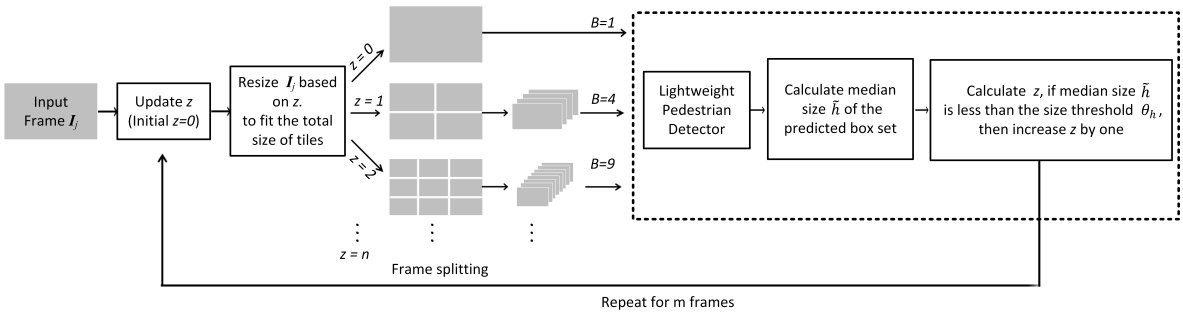

FIGURE 2. Auto-optimizing the zooming settings for pedestrian detection. When $z = 0$ the whole frame will be passed for detection process. When $z > 0$ the frame will be split into $(z + 1)^2$ non-overlapped tiles.

to jointly build a semantic segmentation model and pedestrian detection model with a different strategy. They used semantic segmentation for both proposal generation and prediction refinement.

To improve the fine-grained classification, the input images are divided into set of crops that are processed with CNN separately [30]. In this way, both object-level and part-level attentions are employed. Similarly, but with overlapped crops, the authors in [31] employed recurrent attention for the same purpose.

The pedestrian datasets commonly used in the literature include the TUP-det [15] that was published over ten years ago, the INRIA [27] and ETH [16] that are more comprehensive, and the well-known Caltech Pedestrian Dataset [17] that has become very popular in computer vision community. There are some other datasets summarized in [14]. However, most of these datasets are not captured by surveillance cameras in outdoor scenario.

Nowadays surveillance cameras are widely used and in some application scenarios they are installed from a far distance to cover a wide and long range. As a result, the pedestrians in the videos appear in different sizes. In this paper, we attempt to design a real-time pedestrian detection algorithm which can detect pedestrians, in particular, those of small sizes.

III. PROPOSED FRAMEWORK

As shown in Section II, most of the current algorithms mainly concern the detection accuracy rather than speed. However, in most real-world applications, speed is just as important as accuracy, which motivates us to develop a fast and efficient pedestrian detection framework.

In this section we explain in detail our proposed framework, which features two main parts: the attentive auto-zooming technique and the lightweight CNN-based pedestrian detector. In the starting up stage, the first few input frames are used to automatically determine the optimal scene settings for the framework. Optimization of the zooming settings is achieved based on the distance between the surveillance camera and pedestrian objects. This distance, which can be estimated from the pedestrian size on input frames, determines a proposed zooming factor z. The zooming is then conducted by splitting the input frame into $(z + 1)^2$ non-overlapped tiles. Next, tile selection is conducted in order to keep only the tiles containing pedestrians. The optimal settings obtained in the zooming and tile filtering steps will be used for the rest of the detector.

A. AUTO-ZOOMING-BASED DETECTION

Optimal zooming settings are obtained by running the following steps on a defined number m of input frames. First, we specify the default input frame resolution $H_d \times W_d \times D$, where H_d represents the height, W_d the width and D the channel number respectively. The zooming factor z is initialized as 0, which indicates that no zooming step is applied yet. The first frame I_1 is received and resized to fit the input dimension of the lightweight detector with size $H_d \times W_d \times D$. The resized frame is then fed into the lightweight detector with batch size B as formulated in (1).

$$B = \begin{cases} (z+1)^2, & \text{if } z \geq 1 \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

Next, the zooming factor z will be calculated based on the size of the predicted pedestrians as formulated in (2).

$$z = \begin{cases} z+1, & \text{if } \tilde{h} < \theta_h \\ z, & \text{otherwise} \end{cases} \tag{2}$$

where $\tilde{h}$ is the median height of the predicted bounding boxes and θ_h is the default height threshold. The zooming factor z is then used for next input frame processing. Fig. 2 illustrates the steps applied on input frame based on the zooming factor z.

When $B \geq 4$, which happens when the zooming factor is updated to 1, the input frame will be resized by (3) to fit the total size of all proposed tiles. All tiles share the same resolution $H_d \times W_d \times D$.

$$H = \frac{B}{2} \times H_d, \quad W = \frac{B}{2} \times W_d \tag{3}$$

In the next step for this case, the resized input frame of dimension $H \times W \times D$ is split into B non-overlapped tiles. Then, all tiles are fed to the lightweight pedestrian detector in the batch inference mode. The recalculated median $\tilde{h}$ is used to obtain z for the process of the next frame. This procedure is repeated for m input frames, ending up with an optimal z.

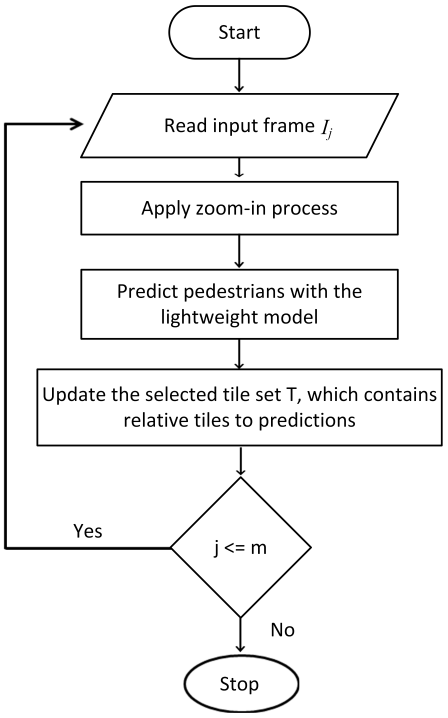

FIGURE 3. Tile selection is run after reaching the best zooming settings. This stage can be run for longer time to avoid missing important tiles.

B. TILE SELECTION

After the auto-zooming process explained in Sect. III-A we get a set of non-overlapped tiles B. Each tile is related to a region in the input frame, which may or may not hit the region of interest, i.e. regions containing pedestrians. To speed up computation, we simply pick those tiles containing pedestrians and drop out the others. Fig. 3 shows the process of finding the optimal tile set T. The tiles obtained from splitting stage are passed into the pedestrian detector, resulting in a set of pedestrians for each tile. However, for some tiles it is unlikely to contain any pedestrian (e.g. sky, see the upper right region of Fig. 1). If no pedestrian is detected in a tile for the first m consecutive frames, this tile will be excluded from subsequent batch inferencing.

C. THE PROPOSED LIGHTWEIGHT PEDESTRIAN DETECTOR

Our lightweight pedestrian detector is a single shot fully convolutional neural network that contains two main parts: feature extractor and pedestrian predictor. The feature extractor consists of one convolutional layer followed by seven sequenced depthwise separable convolutional layers. Fig. 4 illustrates the proposed model and its characteristics.

With the exception of the first layer that uses regular convolution, we employ depthwise separable convolution for the rest layers. Depthwise separable convolution was introduced in [18] and used in Xception [19] and Mobilenet [20]. The proposed model is similar to mobilenet in some aspects such as the number of filters, the depth multiplier, and

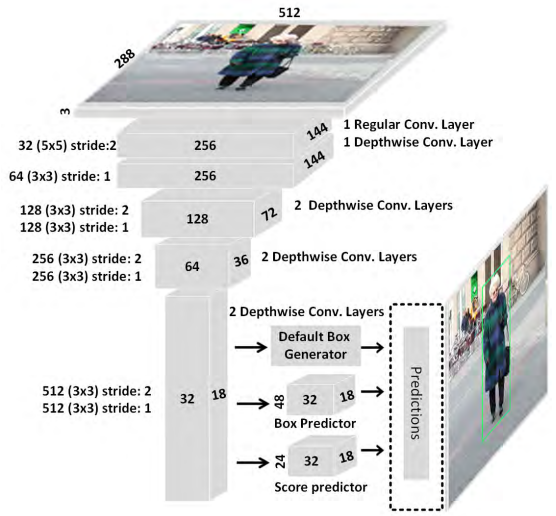

FIGURE 4. Lightweight pedestrian detection model.

the sequence of layer block (zero padding + depthwise convolution+ReLU [33] + batch normalization [32] + pointwise convolution + ReLU + batch normalization). However, it differs from Mobilenet in the number of layers and the kernel size of the first convolutional layer.

In the first regular convolutional layer, we use 32 kernels of size 5×5 in order to force this layer to pay more attention to the generic features rather than the fine details of the pedestrians. By default this convolutional layer is formulated by (4):

$$G_{k,l,n} = \sum_{i,j,m} K_{i,j,m,n} \cdot F_{k+i-1,l+j-1,m} \tag{4}$$

where the m-th kernel K of size $D_K \times D_K \times M \times N$ is applied to the m-th input channel F of size $D_F \times D_F \times M$ to generate the n-th output channel on the output feature map G of size $D_G \times D_G \times N$. The spatial dimension of the output feature map, kernel and input are denoted by D_G, D_k and D_F respectively. In our model, we have used 32 kernels of size 5×5, input image with 3 channels, and output feature map of 32 depth. The stride step in our first layer is 2 pixels.

The convolution process in the next seven depthwise separable convolutional layers is decomposed into depthwise convolution and pointwise convolution [20]. Firstly, it convolves a single 3×3 filter over each input channel separately, then it combines the outputs by pointwise convolution of 1×1 kernels. The depthwise convolution is formulated as (5).

$$\hat{G}_{k,l,m} = \sum_{i,j} \hat{K}_{i,j,m} \cdot F_{k+i-1,l+j-1,m} \tag{5}$$

where the m-th filter of kernel $\hat{K}$ is applied to the m-th input channel F to generate the m-th output feature map $\hat{G}$. Each layer in the feature extractor is preceded by zero-padding and followed by batch normalization [32] and ReLU nonlinearity.

The second part of the proposed lightweight pedestrian detector is similar to SSD [7], which uses two convolutional

layers on top of each feature map to predict the bounding boxes and their corresponding existence scores. However, instead of using several feature maps with different dimensions, our model utilizes a single feature map. Over each cell of this feature map, a set of 1×1 kernels instead of 3×3 are applied in order to predict a set of 12 bounding box offsets.

The default bounding box generator is employed to generate an equivalent number of boxes predicted by the model. For each cell over the output feature map, a set of 12 different sized boxes is generated. All default box sizes $S = \{s1, s2, , , s12\}$ share the same default pedestrian aspect ratio $A = 0.41$.

Comparing to Mobilenet, we have designed a shallow CNN feature extractor with only 8 layers due to the problem nature of pedestrian detection. The CNN does not need to pay attention to the fine-grained details of pedestrians because all pedestrian objects are semantically similar. In addition, some low level features are not needed in pedestrian detection such as clothes colors. Along with the mentioned factors, the proposed predictor structure and the zooming technique lead to an efficient performance of the lightweight pedestrian detector.

D. MODEL PARAMETERS

The first convolutional layer of our lightweight detector is the only layer which consumes a large computing time comparing to other depthwise separable convolutional layers. The cost of this layer can be calculated by $D_K \times D_K \times M \times N \times [\frac{D_F}{s} \times \frac{D_F}{s}]$, where s is the stride. For our default setting with input dimension 512×288 pixels and stride $s = 2$, the computational cost of this layer is 88.47M macs. For depthwise separable layers, the total computational cost of each combination of depthwise convolution and pointwise convolution is computed by $D_K \times D_K \times M \times \frac{D_F}{s} \times \frac{D_F}{s} + M \times N \times D_F \times D_F$. Overall, our lightweight model costs 903.198M macs with only 569, 792 parameters for the default input dimension 512×288.

In general, our lightweight model size is 2.3 MB and consumes about 400 MB memory in running time. It is able to process more than 160 fps on GTX 1080 Ti.

E. TILE STITCHING AND BOUNDING BOX POST-PROCESSING

The post-processing involves tiles stitching as well as reallocating each predicted bounding box in its relative spatial position over the original input frame. However some bounding boxes are shared by two tiles horizontally or vertically. We added a small procedure to connect those bounding boxes on different tiles of the same pedestrian object. Fig. 5 shows two samples of pedestrians shared by two tiles. In the left sample, both pedestrians heads are detected on the upper tile while their lower body are detected on the lower tile. For the same pedestrian object, we compose a bigger box that covers both of the two separated boxes, i.e., the parameters of the new box are calculated by (6).

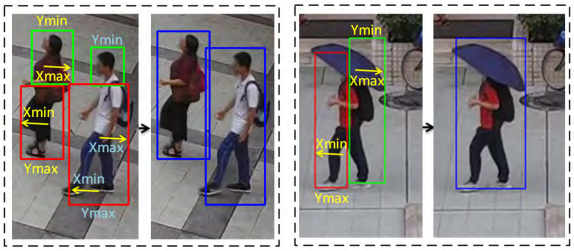

FIGURE 5. Bounding box post-processing. The boxes with green and red colors show the pedestrians that are shared over two neighboring tiles, whereas the pedestrian bounding boxes after post processing are shown in blue color.

$$\begin{aligned} Xmin &= Min(Xmin1, Xmin2) \\ Ymin &= Min(Ymin1, Ymin2) \\ Xmax &= Max(Xmax1, Xmax2) \\ Ymax &= Max(Ymax1, Ymax2) \end{aligned} \tag{6}$$

The bounding boxes after post-processing are shown with blue color in Fig. 5.

An illustration of the inference phase of the proposed framework is shown in Fig. 6. The optimal settings in this example are $z = 1$ and selected tile set $T = \{t1, t2, t3, t4\}$, which means all the regions in the frame are taken into account.

IV. TRAINING

A. OUTDOOR PEDESTRIAN DATASET

The lack of public surveillance datasets for pedestrian detection with small objects motivated us to collect and annotate an outdoor surveillance pedestrian dataset. Our dataset features pedestrians that are really small, about half the size of pedestrians in Caltech dataset. The videos are captured outdoor in different lighting conditions and at different times of the day. This dataset attempts to reflect the various challenges met in real-world surveillance videos. The captured videos are then grouped into one training set and two testing sets. The training set and the first testing set contain the regular small size pedestrians, while the second testing set contains the more challenging videos for very far pedestrians. For the training set and the regular testing set (jointly referred to as the "main set" in the rest of the paper), we picked up one frame for every four seconds, i.e. 1/4 fps, while for the challenging testing set, one frame is picked up for every six seconds, i.e. 1/6 fps. This procedure could minimize object redundancy to ensure that we acquire a diversity of pedestrian samples. Fig. 7 gives some frame samples from both the main set and the challenging set. Fig. 8 shows the histograms of the pedestrian size and aspect ratio of the main and challenging sets respectively. The statistics of this dataset is summarized in Table 1.

B. TRAINING LIGHTWEIGHT PEDESTRIAN DETECTOR

On the dataset described in Sect.IV-A, we have trained our lightweight pedestrian detection model (shown in Fig. 4)

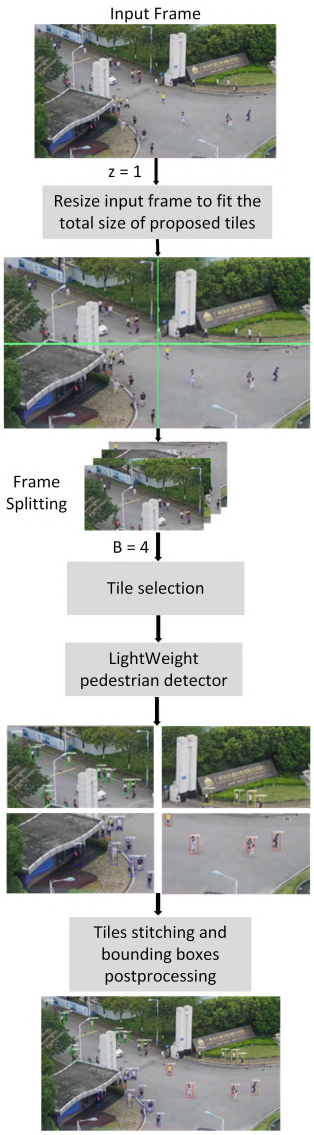

FIGURE 6. An illustration of the inference phase of the proposed framework with zooming factor $z = 1$ and selected tile set $T = \{t1, t2, t3, t4\}$.

with Caffe framework [34] version 1.0.0-rc3, on Nvidia GTX 1080 Ti. We set the 12 default scales to be generated for each cell on the output feature map to $S = \{0.02, 0.04, 0.10, 0.17, 0.24, 0.31, 0.38, 0.45, 0.52, 0.59, 0.66, 0.73\}$ with the static default aspect ratio $A = 0.41$. Our feature extractor is fine-tuned with the classification network Mobilenet [20], which is pretrained on Imagenet [28]. A set of random image augmentations on input images is applied including the cropping, brightness and color jittering. From the training set, the mean values of the red, green and blue color channels were calculated. For each frame, the mean values are subtracted from the corresponding channels. We have trained our lightweight model for 300K iterations with batch size 32. The optimizer RMSprop [22] is used with an initial learning rate of 0.01, which is multiplied by 0.1 for every 100K iterations.

TABLE 1. Statistics of the utilized outdoor surveillance pedestrian dataset.

Set	# Frames	# Pedestrians	Size Median	FPS
Train	12,649	60,881	124	0.25
Test 1	3,308	17,970	117	0.25
Test 2	3,771	49,164	59	0.16

Similar to Faster R-CNN [3] and SSD [7], Jaccard overlap is used for matching the generated boxes to the received ground truth boxes as an input. Then the higher scored default boxes are filtered by an overlap threshold (by default 0.5). Finally, the non-maximum suppression (NMS) is used to resolve the conflicts when several predicted bounding boxes overlap. In our model, if two boxes are overlapped by more than 45%,then only the one with the highest score is kept.

The objective function used in this model is similar to Faster R-CNN, MultiBox [21] and SSD, but with some modifications. For score prediction we used softmax classifier over presence and absence of pedestrian (pedestrian and background), as formulated in (7).

$$L_{conf}(x, c) = -\sum_{i \in Pos} x_{ij} log(\hat{c}_i) - \sum_{i \in Neg} log(\hat{c}_i^0)$$
$$where \quad \hat{c}_i = \frac{e(c_i)}{\sum e(c_i)} \tag{7}$$

When the i-th default box matched the j-th ground truth box with overlapped region higher than 0.5 (using Jaccard overlap), then $x_{i,j} = 1$. To balance the number of negative and positive predictions, we only use the highest scored part of the negative set. Following the settings of SSD in [7], we keep the ratio as 3 negative to 1 positive sample.

For the localization loss we used Smooth $L1$ loss [2] between the i-th predicted offset of a box l^m and each j-th corresponding ground truth box coordinate $\hat{g}^m$ as in [7], which is formulated by (8).

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij} smooth_{L_1}(l_i^m - \hat{g}_j^m)$$
$$\hat{g}_j^{cx} = (\hat{g}_j^{cx} - \hat{d}_i^{cx})/\hat{g}_i^w, \quad \hat{g}_j^{cy} = (\hat{g}_j^{cy} - \hat{d}_i^{cy})/\hat{g}_i^h,$$
$$\hat{g}_j^w = log(\hat{g}_j^w/\hat{d}_i^w) \quad \hat{g}_j^h = log(\hat{g}_j^h/\hat{d}_i^h)$$
$$smooth_{L_1}(b) = \begin{cases} 0.5b^2, & \text{if } |b| < 1 \\ |b| - 0.5, & \text{otherwise} \end{cases} \tag{8}$$

where m represents the box coordinates while (cx, cy), w, h denote the box center, width, and height respectively. The final loss function is formulated as the weighted sum of the localization loss L_{loc} and the confidence loss L_{conf} as in (9).

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \tag{9}$$

Localization loss L_{loc} is weighted with the term α which is set by default to 1 and the whole weighted loss is normalized by the number of default boxes N.

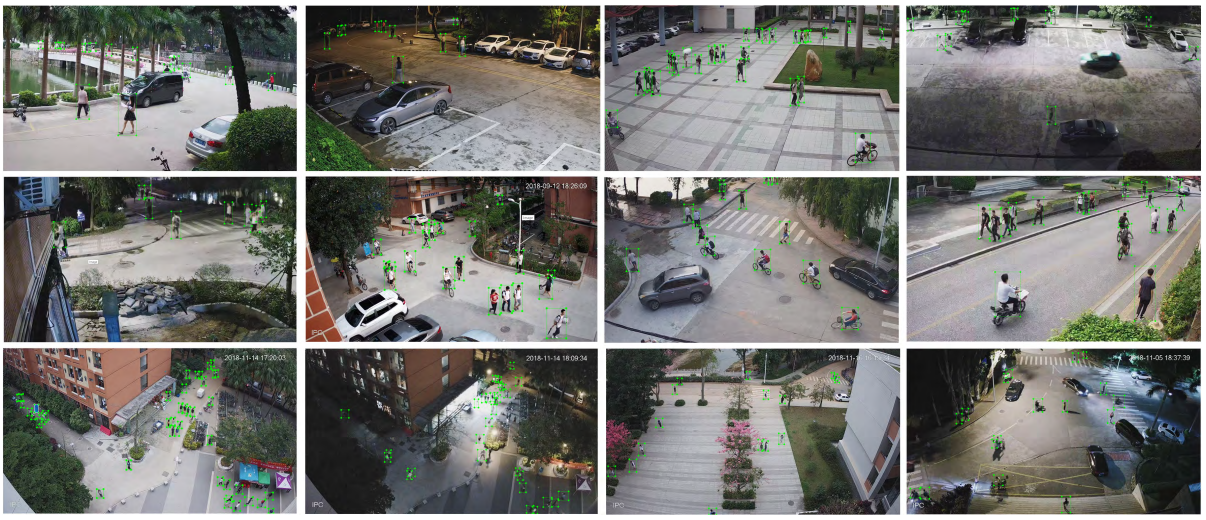

FIGURE 7. Outdoor surveillance pedestrian dataset. The first two rows show some samples of the main regular set, while the last row shows some samples from the challenging testing set.

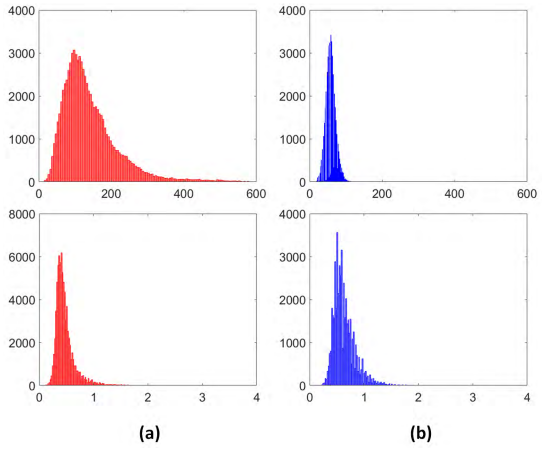

FIGURE 8. The left column (a) shows in red color the histogram distribution of the pedestrian objects size (height), and the histogram distribution of the aspect ratio in the main set with frame size of 1920 × 1080, whereas the right column (b) shows in blue color the histogram distributions of pedestrian objects size and aspect ratio in the challenging set respectively.

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

We have conducted intensive experiments to evaluate our framework components as well as evaluating the framework against other customized models.

A. SPATIAL RESOLUTION AND ASPECT RATIO

We began by evaluating the effects of aspect ratios on training and detection performance. We have trained two instances of our lightweight model, one with aspect ratio 1:1 and input dimension 384 × 384 pixels and the second model with input dimension 512 × 288 pixels and the most common aspect ratio in modern surveillance cameras (16:9). Both instances have the same totaled spatial resolution of 147, 456 pixels. We have trained both models for the same number of iteration 300K with batch size 32. It is found that the aspect ratio has a significant impact on pedestrian detection. The network with 16:9 aspect ratio not only learns better in training but also performs better in the inference phase. Table 2 summarizes the characteristics of both models and their performance. Fig. 9 shows the plot of Precision-Recall (PR) and ROC plot of miss-rate/false positive per image (FPPI).

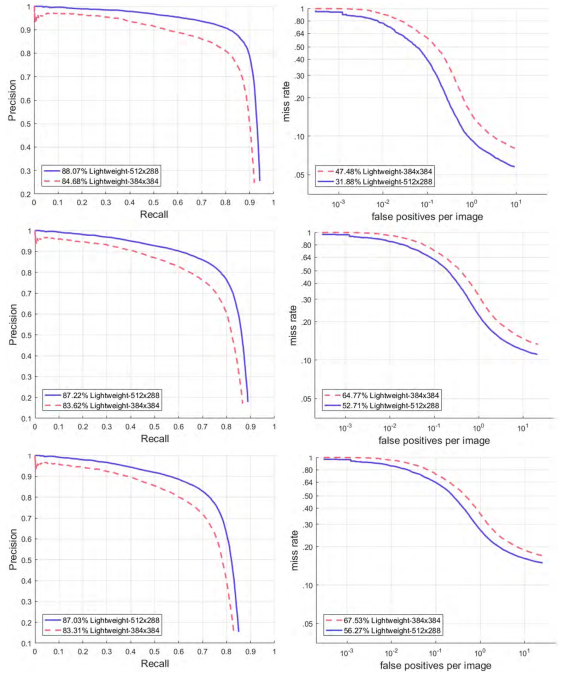

FIGURE 9. Influence of aspect ratio of model input on pedestrian detection performance. The original video dimension is 1920 × 1080. The first,second and last rows show the PR plots and miss rates against FPPI for Large-scale, Reasonable-scale and All-scale respectively. The scale metrics employed here are defined in [17].

B. ZOOMING TECHNIQUE EVALUATION

In this part, we evaluate our model with and without zooming step on the challenging set. Which contains more challenging images (far pedestrians). Although it would compromise

TABLE 2. Performance of twin pedestrian detectors with different aspect ratios.

Aspect Ratio	Input Dimension	AP			FPPI			FPS
		Large	Reasonable	All	Large	Reasonable	All	
1:1	384 × 384	84.68	83.62	83.31	47.48	64.77	67.53	161
16:9	512 × 288	88.07	87.22	87.03	31.88	52.71	56.27	161

TABLE 3. Lightweight detector performance against several customized single-shot detectors.

Model	Input Dimension	AP			FPPI			FPS
		Reasonable	All	Near	Reasonable	All	Near	
SSD300-VGG16	300 × 300	85.19	84.9	58.96	63.52	65.96	52.55	74
SSD512-Mobilenet v1	512 × 512	85.24	84.86	86.00	63.92	66.37	55.59	62
YOLO v3 light Mobilenet v1	416 × 416	86.58	86.23	87.58	59.61	62.23	48.26	87
YOLO v3 Mobilenet v1	416 × 416	87.05	86.80	87.73	56.94	59.75	45.28	83
Lightweight-512x288(ours)	512 × 288	**87.22**	**87.03**	**87.74**	**52.71**	**56.28**	**40.99**	**161**

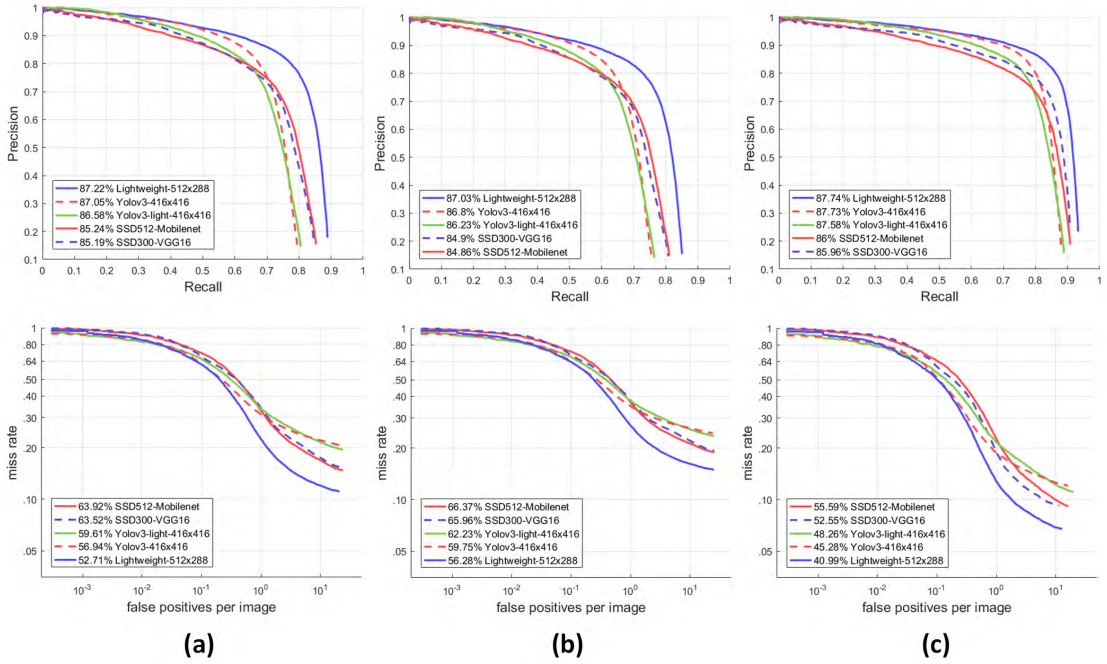

FIGURE 10. Performance of our lightweight pedestrian detector against several single-shot customized object detectors. The first row shows PR plots of Reasonable, All and Near scales respectively, while the second row shows ROC plots.

TABLE 4. Performance of our zooming-based framework against Faster R-CNN on the challenging testing set.

Model	AP	FPPI	FPS
Faster RCNN	47.20	96.12	17
Lightweight-512x288 z=0	48.05	96.60	**161**
Lightweight-512x288 z=1	**77.70**	**87.33**	42

some speed in each zooming step, a significant improvement is obtained on surveillance videos in terms of detection rate for small pedestrians. In order to test the zooming impact, we have disabled the auto-zooming mechanism and set the zooming factor z manually to 0 and 1. In Table 4 we summarized the performance of our lightweight pedestrian detector with and without zooming. The Precision/Recall and Miss Rate/FPPI plots are shown in Fig. 11(b) while the speed against AP performance is shown in Fig.12(b).

C. LIGHTWEIGHT DETECTOR PERFORMANCE AGAINST CUSTOMIZED SINGLE-SHOT DETECTORS

The lightweight pedestrian detector is evaluated against two customized instances of SSD [7] (SSD300-VGG16 and SSD512-MobilenetV1) due to the similarities on training and loss calculation between SSD model and ours. We fined-tuned the SSD's predictor for tiny objects. Specifically, we have made all aspect ratios smaller than 1 to fit the pedestrian shape. (The aspect ratio in the original SSD paper ranges between 0.5 and 1.5). We have also made the scale of boxes smaller (between 20 and 250 pixels) than the original form (between 60 to 264 pixels). We have trained SSD300-VGG16 [7] with 300 × 300 dimension and SSD-Mobilenet-v1 with 512 × 512 dimension. We use 8 default bounding boxes per cell in each feature map of the total 6 feature maps.

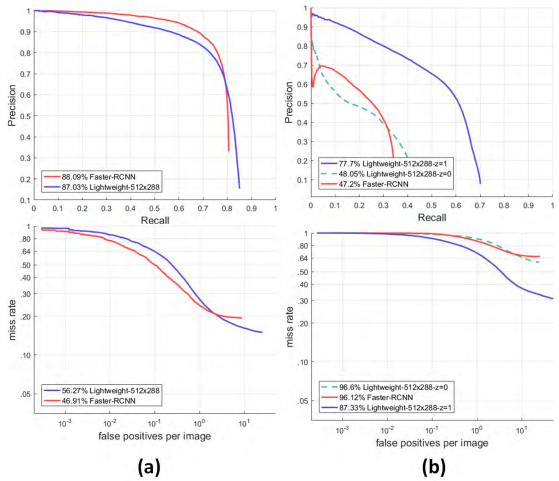

FIGURE 11. Performance of the zooming-based lightweight predictor against two-stage pedestrian detector Faster R-CNN. The left column (a) shows our default non-zoomed model performance against Faster R-CNN on the main testing set and the right column (b) shows performance of our zoomed model with one/two steps against Faster R-CNN.

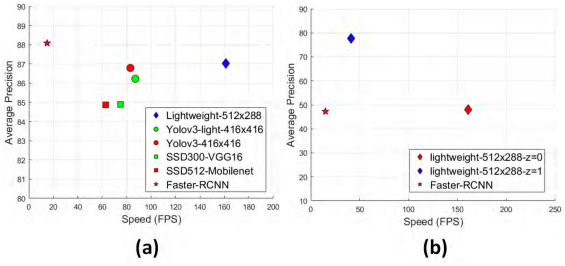

FIGURE 12. Forward speed in frame per second against average precision (AP/fps), the left plot (a) shows the comparison of single-shot models on main testing set, while the right plot (b) shows our framework against Faster R-CNN speed on the challenging testing set.

For YOLO, we have adopted the variations YOLO version 3 [35] and YOLO-v3-light [35] since they perform better than the standard YOLO. We followed most default settings of YOLO v3 and YOLO-v3-light and set the number of filters in the last layer as 18 for our two class pedestrian detection problem. We used default Mobilenet v1 [20] as the baseline network with input size 416×416. The pretrained networks VGG16 and Mobilenet v1 on Imagenet [28] are used to fine-tune their corresponding detectors.

There are two main reasons to choose these models to evaluate against our lightweight model. Firstly, our proposed framework is flexible and can be applied for some other general object detection. Secondly, these evaluated object detectors are the state-of-the-art in terms of speed. We have summarized the complexity of the models in terms of number of parameters and the computational cost (Mult-Adds) in Table 5. Fig. 13 shows the size ratio of each base network to its corresponding predictors. The computational cost of these models is shown in Fig. 14.

From Fig. 10, it is clear that our lightweight detector achieves the best performance in terms of Precision/Recall and Miss Rate/FPPI. Further details of the used models performance and characteristics are listed in Table 3.

D. ZOOMING LIGHTWEIGHT AGAINST TWO-STAGE PEDESTRIAN DETECTOR

Although the zooming technique improves the detection accuracy, it sacrifices the speed to some extent. That pushed us to evaluate our model against the two-stage object detector (Faster R-CNN). We have trained Faster R-CNN with VGG16 [24] as the base network. Table 4 summarizes the performance of our zooming-based lightweight with zooming steps $z = 1$ against Faster R-CNN. While Fig. 11.(a) shows a good performance of Faster R-CNN on the main set, Fig. 11.(b) shows that our framework is superior to Faster R-CNN on the challenging testing set.

Our zooming-based detection mostly depends on the acquired frame resolutions, where a large input frame dimension improves the prediction process in both scoring

TABLE 5. Comparison of the models in terms of number of parameters and the computational cost (Mult-Adds).

Model	Baseline Net	Input Dim	Baseline Params	Predictor Params	Total Params	Mult-Adds (Billion)
Faster R-CNN	VGG16	224x224	20,133,504	116,556,496	136.690,000	51.69
SSD	VGG16	300x300	20,133,504	3,259,392	23,392,896	30.43
YOLO-v3	Mobilenet v1	416x416	3,185,088	6,472,360	9,657,448	5.018
SSD	Mobilenet v1	512x512	3,185,088	2,381,824	5,566,912	2.32
YOLO-v3-Light	Mobilenet v1	416x416	3,185,088	1,906,688	5,091,776	2.40
Lightweight (ours)	Tiny-net (8 layers)	512x288	532,928	36,864	569,792	0.90

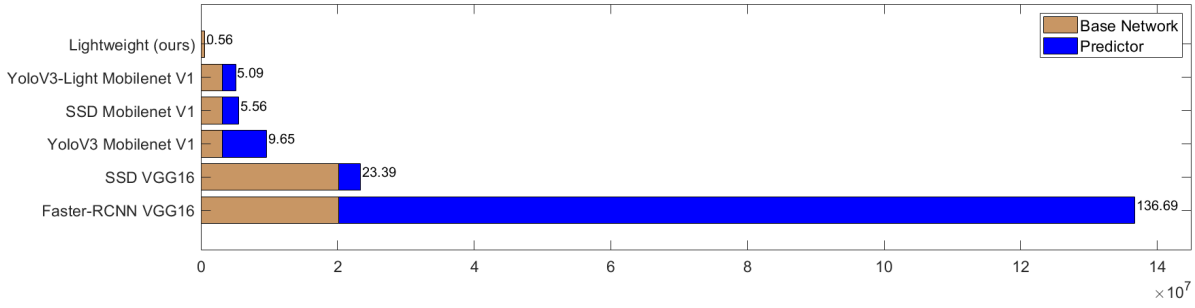

FIGURE 13. Size ratio of the baseline network to predictor in terms of number of parameters (in million).

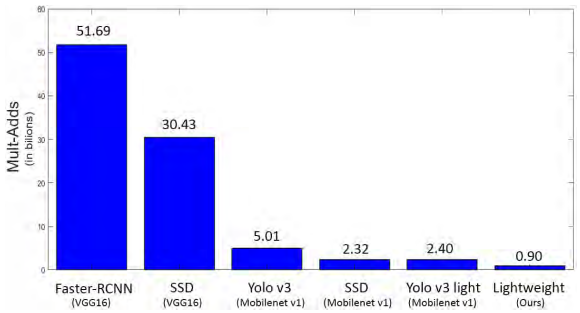

FIGURE 14. The computational complexity of the evaluated detectors in billions of Mult-Adds.

and localization. When we apply extra zooming step $z = 2$ on the challenging testing set we observed that the obtained performance is degraded due to the limit of input frame resolution. But when we apply it in real scenario with high resolution such as that in Fig. 1, the obtained performance is improved by applying more than one zooming step.

VI. CONCLUSION

In this paper we introduced a novel framework for outdoor surveillance video analysis, specifically for far pedestrian detection. We have devised an auto-zooming-based detection technique to improve the detection performance in terms of AP and FPPI. Moreover, we significantly improved the detection speed by designing a lightweight CNN-based model for real-time pedestrian detection. The proposed framework efficiently solved the small (far) pedestrians detection task on which most of the existing surveillance systems are struggling.

Moreover, the proposed virtual auto-zooming technique can be applied to both general and customized object detection solutions, which is our plan for future work.

ACKNOWLEDGMENT

The authors would like to thank the graduate students: Xiangquan Chen, Xin LI, Yifei Gao, Yuyi Lin, Zeqiong Yu, Xin Luo, Kun Zhang, Qunqi Zeng, and Runze Chen for their contributions on annotating the utilized dataset in this work. Finally, they thank the anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[2] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1440–1448.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[4] W. Ouyang, X. Wang, X. Zeng, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, C.-C. Loy, X. Tang, "DeepID-Net: Deformable deep convolutional neural networks for object detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2403–2412.

[5] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, "Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,"İin *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2874–2883.

[6] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, "OverFeat: Integrated recognition, localization and detection using convolutional networks," 2013, *arXiv:1312.6229*. [Online]. Available: https://arxiv.org/abs/1312.6229

[7] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, *SSD: Single Shot Multibox Detector* (Lecture Notes in Computer Science), vol. 9905. 2016, pp. 21–37.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[9] W. Ouyang and X. Wang, "Joint deep learning for pedestrian detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2056–2063.

[10] X. Jiang, Y. Pang, X. Li, and J. Pan, "Speed up deep neural network based pedestrian detection by sharing features across multi-scale models," *Neurocomputing*, vol. 185, pp. 163–170, Apr. 2016.

[11] D. Ribeiro, J. C. Nascimento, A. Bernardino, and G. Carneiro, "Improving the performance of pedestrian detectors using convolutional learning," *Pattern Recognit.*, vol. 61, pp. 641–649, Jan. 2017.

[12] D. Varga and T. Szirányi, "Detecting pedestrians in surveillance videos based on convolutional neural network and motion," in *Proc. 24th Eur. Signal Process. Conf. (EUSIPCO)*, Budapest, Hungary, Aug./Sep. 2016, pp. 2161–2165.

[13] X. Wang, M. Wang, and W. Li, "'Scene-specific pedestrian detection for static video surveillance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 2, pp. 361–374, Feb. 2014.

[14] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, Apr. 2012.

[15] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[16] A. Ess, B. Leibe, and L. Van Gool, "Depth and appearance for mobile scene analysis," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.

[17] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 304–311.

[18] L. Sifre, "Rigid-motion scattering for image classification," Ph.D. dissertation, Ecole Polytechn., CMAP, Paris, France, 2014.

[19] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1251–1258.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: https://arxiv.org/abs/1704.04861

[21] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2147–2154.

[22] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Netw. Mach. Learn.*, vol. 4, no. 2, pp. 26–31, 2012.

[23] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *Computer Vision—ECCV* (Lecture Notes in Computer Science), vol. 9908, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016.

[24] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[25] X. Du, M. El-Khamy, J. Lee, and L. Davis, "Fused DNN: A deep neural network fusion approach to fast and robust pedestrian detection," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 953–961.

[26] L. Zhang, L. Lin, X. Liang, and K. He, *Is Faster R-CNN Doing Well for Pedestrian Detection?* (Lecture Notes in Computer Science), vol. 9906. 2016, pp. 443–457.

[27] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection,"İin *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 886–893.

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[29] C. Lin, J. Lu, and J. Zhou, "Multi-grained deep feature learning for pedestrian detection," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2018, pp. 1–6.
[30] Y. Peng, X. He, and J. Zhao, "Object-part attention model for fine-grained image classification," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1487–1500, Mar. 2018.
[31] J. Liang, J. Guo, X. Liu, and S. Lao, "Fine-grained image classification with Gaussian mixture layer," *IEEE Access*, vol. 6, pp. 53356–53367, 2018.
[32] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, Lille, France, pp. 448–456, 2015.
[33] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 807–814.
[34] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, vol. 14, Nov. 2014, pp. 675–678.
[35] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: https://arxiv.org/abs/1804.02767

SAGHIR ALFASLY received the B.S. degree in computer information systems from Sana'a University, Yemen, in 2010, and the M.Sc. degree in computer science from Kuvempu University, India, in 2015. He is currently pursuing the Ph.D. degree in information and communication engineering with the School of Electronic and Information Engineering, South China University of Technology, China. His research interests include surveillance image and video analysis, person/vehicle re-identification, computer vision, and machine learning.

BEIBEI LIU received the Ph.D. degree in communication and information system from Sun Yat-sen University, China, in 2009. She has been a Researcher with the Korean Advanced Institute of Science and Technology, and Newcastle University, U.K. She is currently an Assistant Professor with the School of Electronic and Information Engineering, South China University of Technology. Her research interests include multimedia information security and machine learning.

YONGJIAN HU received the Ph.D. degree in communication and information systems from the South China University of Technology, in 2002, where he is currently a Full Professor with the School of Electronic and Information Engineering. From 2011 to 2013, he was a Marie Curie Fellow with the Department of Computer Science, University of Warwick, U.K. From 2006 to 2008, he was a Research Professor with the Department of Computer Science, Korea Advanced Institute of Science and Technology (KAIST), South Korea. From 2005 to 2006, he was a Research Professor with the School of Information and Communication Engineering, SungKyunKwan University, South Korea. From 2000 to 2004, he was with the Department of Computer Science, City University of Hong Kong, four times as a Research Assistant, a Senior Research Associate, and a Research Fellow, respectively. He has published more than 70 peer-reviewed papers. His research interests include information hiding, multimedia security, and machine learning. He is a Senior Member of the Chinese Institute of Electronics (CIE) and a Senior Member of the China Computer Federation (CCF).

YUFEI WANG received the B.E. degree in information engineering from the South China University of Technology, in 2010, and the Ph.D. degree in information and communication engineering from the South China University of Technology, in 2018. He is currently a Research Fellow with the Sino-Singapore International Joint Research Institute. His research interests include information forensics, steganography and steganalysis, computer vision, and machine learning.

CHANG-TSUN LI received the B.Sc. degree in electrical engineering from National Defence University (NDU), Taiwan, in 1987, the M.Sc. degree in computer science from U.S. Naval Postgraduate School, Monterey, CA, USA, in 1992, and the Ph.D. degree in computer science from the University of Warwick, U.K., in 1998. He was an Associate Professor with the Department of Electrical Engineering, NDU, from 1998 to 2002, and a Visiting Professor with the Department of Computer Science, U.S. Naval Postgraduate School, in the second half of 2001. He was a Professor with the Department of Computer Science, University of Warwick, U.K., until January 2017, and a Professor of Charles Sturt University, Australia, from January 2017 to February 2019. He is currently a Professor with the School of Information Technology, Deakin University, Australia. His research interests include multimedia forensics and security, biometrics, data mining, machine learning, data analytics, computer vision, image processing, pattern recognition, bioinformatics, and content-based image retrieval. The outcomes of his multimedia forensics research have been translated into award-winning commercial products protected by a series of international patents and have been used by a number of police forces and courts of law around the world. He is currently an Associate Editor of the *EURASIP Journal of Image and Video Processing* (JIVP) and an Associate Editor of the *IET Biometrics*. He involved in the organization of many international conferences and workshops and also served as a member of the international program committees for several international conferences. He is also actively contributing keynote speeches and talks at various international events.