Computers, Materials & Continua
DOI: 10.32604/cmc.2023.035974
Article

Tech Science Press

A Deep Learning-Based Crowd Counting Method and System Implementation on Neural Processing Unit Platform

Yuxuan Gu, Meng Wu*, Qian Wang, Siguang Chen and Lijun Yang

Nanjing University of Posts and Telecommunications, Nanjing, P210023, China
*Corresponding Author: Meng Wu. Email: wum@njupt.edu.cn
Received: 12 September 2022; Accepted: 12 November 2022

Abstract: In this paper, a deep learning-based method is proposed for crowd-counting problems. Specifically, by utilizing the convolution kernel density map, the ground truth is generated dynamically to enhance the feature-extracting ability of the generator model. Meanwhile, the "cross stage partial" module is integrated into congested scene recognition network (CSRNet) to obtain a lightweight network model. In addition, to compensate for the accuracy drop owing to the lightweight model, we take advantage of "structured knowledge transfer" to train the model in an end-to-end manner. It aims to accelerate the fitting speed and enhance the learning ability of the student model. The crowd-counting system solution for edge computing is also proposed and implemented on an embedded device equipped with a neural processing unit. Simulations demonstrate the performance improvement of the proposed solution in terms of model size, processing speed and accuracy. The performance on the Venice dataset shows that the mean absolute error (MAE) and the root mean squared error (RMSE) of our model drop by 32.63% and 39.18% compared with CSRNet. Meanwhile, the performance on the ShanghaiTech PartB dataset reveals that the MAE and the RMSE of our model are close to those of CSRNet. Therefore, we provide a novel embedded platform system scheme for public safety pre-warning applications.

Keywords: Crowd counting; CSRNet; dynamic density map; lightweight model; knowledge transfer

1 Introduction

Crowd counting of public areas (such as commercial districts and tourist attractions) is indispensable in risk prediction and maintaining public order. With the rapid development of deep learning, crowd-counting tasks can generally be handled with approaches based on object detection or density maps. The former solution obtains the counting results with the help of object detection networks such as You Only Look Once v4 (YOLOv4) [1] and Single Shot Multibox Detector (SSD) [2], which shows considerable performance under the circumstance with a sparse crowd in sight. However, it may show deviations in the detection results handling dense scenes due to people occlusion.

Therefore, it is becoming a tendency to tackle the crowd counting problem by estimating the density map of the scene. Zhang et al. [3] coped with the problem of variable head size due to capturing people at different distances by designing a three-column network multi-column convolutional neural network (MCNN), in which each column adopted convolution layers with different kernel sizes. Li et al. [4] integrated the feature extraction backbone network (i.e., congested scene recognition network (CSRNet)) with several dilated convolution layers forming a single-column network to reduce the model parameters. Yan et al. [5] proposed a perspective-guided convolution network (PGCNet) with perspective-guided convolution aiming to smooth the dramatic intra-scene scale variation of the crowd. Thanasutives et al. [6] proposed a modified spectrum-aware feature augmentation network (M-SFANet) based on SFANet [7]. Atrous spatial pyramid pooling [8] and context-aware module [9] were attached to M-SFANet, which enhances extracting multi-scale features and adaptively encodes the scale of the contextual information. Bai et al. [10] designed an adaptive dilated self-correction network (ADSCNet), in which the authors proposed the adaptive dilated convolution to replace convolution with discrete and static dilation rate, and they also constructed a self-correction supervision framework to optimize the model. Scale-adaptive selection network (SASNet) in [11] adopted proper patches to calculate the counting result according to its feature selection strategy, which alleviates the inconsistency problem between the training target and the evaluation metric by training with “pyramid region awareness” loss.

In crowd counting, previous researchers paid more attention to estimating the density map more accurately than exploring methods of generating ground-truth labels. Generally, the method of generating ground-truth labels employs a Gaussian convolution kernel to convolve an annotated picture, in which the position of each head has been marked as a dot. In MCNN [3], it held that the head size is subject to the distance between the target and the center of K persons adjacent to the marked head in a crowded scene. Therefore, it convolved the dot map to obtain the ground-truth density map, and the variance of the kernel was adaptively determined by the average distance between the marked position and its K adjacent ones. In contextual pyramid (CP)-CNN [12], every annotated position was convolved with a fixed-variance kernel, and it dealt with the disappearance of results close to the capturing camera that occurred in MCNN. Oghaz et al. [13] proposed a content-aware density map generation method, which located the nearest head from the annotated dot with the “brute-force nearest neighbor” algorithm and calculated the Gaussian kernel size according to the size of adjacent heads after segmenting with the “Chan-Vese” algorithm.

Methods previously mentioned generated density maps by utilizing Gaussian kernels with fixed bandwidth or adaptive ones where the hyperparameters were selected manually. Wan et al. [14] proposed an adaptive density map generation, and they designed a neural network-based “adaptive density map generation” method. It is an end-to-end training strategy that aims to simultaneously optimize the density map generation network and the crowd counting network. Afterward, they extended their work and designed a kernel-based density map generation (KDMG) framework in work [15], which reduces the workload and takes effect in other object counting fields.

In addition, researchers have shown that knowledge transfer technology helps lightweight networks learn complex feature information and consequently improve their performance. Generally, model performance will degrade while the model size is compressed. However, knowledge transfer is capable of transferring knowledge from excellent models with complex structures to a lightweight network, which compensates for model performance degradation. In image classification, the work [16] succeeded transferring knowledge from a complex model to a simplified one. Zhang et al. [17] deployed the method above in the field of human pose estimation that supervised the hidden layers and output of the student network based on the teacher’s results. For crowd counting, the feasibility of pixel-wise tasks like human pose estimation is proved by researchers. MobileCount [18] generated density maps from hidden layers to optimize the loss between them and corresponding layers in a complex network. Liu et al. [19] achieved "structured knowledge transfer (SKT)" by supervising the patterns and correlations between the layers of the teacher network and the student network. The work [20] proposed a task-specific knowledge distillation framework ShuffleCount to prevent negative knowledge from being transferred to the student network through hierarchic feature regulation.

In this paper, we adopt estimating the density map to achieve crowd counting and select CSRNet as the backbone network of crowd counting for edge devices, which maintains the balance between model size and accuracy. Meanwhile, since knowledge transfer has been successfully employed in pixel-wise tasks (including crowd-counting), the SKT training framework is adopted to transfer complex learned knowledge from teacher to student in an end-to-end manner, and it can enhance the analyzing ability of the lightweight network.

Contributions of this paper are as follows:

(1) Instead of using ground-truth labels with traditional methods, we dynamically generate ground-truth density maps with the image and annotation map as inputs based on the KDMG framework, which reduces the workload sufficiently.

(2) CSRNet is selected as the crowd-counting backbone network. To deploy the model on embedded devices, the "cross stage partial (CSP)" module proposed in CSPNet [21] is introduced to the model, which further reduces the model parameters.

(3) To avoid the deterioration of model performance caused by the reduction of model parameters, the SKT framework is integrated with model training to compensate for the accuracy loss.

(4) After obtaining the lightweight model from the training platform, it is deployed on the TB-RK3399ProD platform, which implements an efficient crowd-counting system.

The remainder of this paper is organized as follows. Section 2 describes our improved crowd-counting scheme. The performance analysis of our scheme is presented in Section 3. In Section 4, we introduce our implementation on the embedded platform. Finally, conclusions are drawn in Section 5.

2 Improved Crowd-Counting Scheme

The improved crowd-counting scheme is divided into three parts: dynamic ground-truth density map generation, lightweight CSRNet and online knowledge transfer.

2.1 Dynamic Ground-Truth Density Maps Generation

At present, estimating density map has become one of the mainstream solutions for crowd counting tasks [3], where the system runs inference on the captured image and considers the sum of pixels in the corresponding density map as the counting result of the scene. Visualizations of object detection-based and density map-based solutions are shown in Figs. 1 and 2.

Before training crowd counting models, it takes an annotated map as input and convolves the input with fixed kernel size to obtain the corresponding density map, which corresponds to the traditional ground-truth label generation. The parameters related to the Gaussian kernel are manually adjusted. Therefore, there may be inconsistency in the head size of different positions, which potentially resulting in low-quality ground-truth labels for training the model.

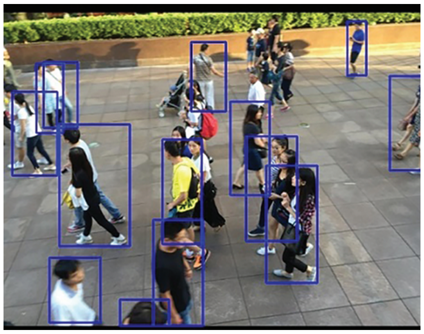
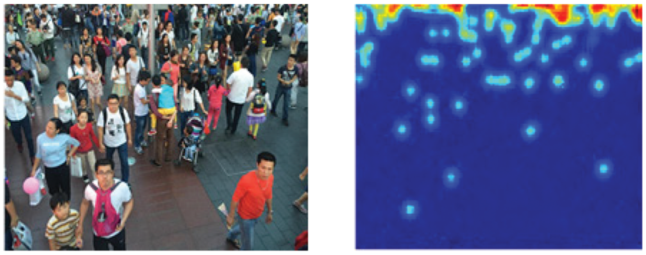


Figure 1: Object detection based crowd counting



(a) Crowd in scene

(b) Density map

Figure 2: Density map-based crowd counting

Given an annotated map D, each annotated point with the value "1" marks one head in the picture, and the density map Y can be represented by formula (1):

$$\mathrm{Y} = \mathrm{D} * \mathrm{k}_\sigma, \tag{1}$$

where k_σ denotes a Gaussian kernel with the bandwidth σ, and "$*$" denotes a convolution. Then, the generated density maps are used as static ground-truth labels for model optimization.

Since the KDMG framework showed its superior performance in object counting tasks in an end-to-end manner, this paper dynamically generates ground-truth density maps for training models with the foundation of the KDMG framework. Based on KDMG, we abandon the last three layers of its original generator model and extract the input images to obtain feature maps, so the resolution of feature maps is consistent with the output of the crowd counting model, which is 1/8 of the input image. Afterward, taking the feature maps as input, the next convolution layer outputs feature maps with the shape of $k^2 \times h \times w$, where the channel denotes the kernel size corresponding to each annotated point. Each annotated point is represented by a $k \times k$ matrix (the annotated point as the center) in the density map. Each matrix is obtained from the output of the convolution layer above reshaped in the channel dimension. Hence, the "efficient channel attention (ECA)" module [22] is added after the last layer to help the kernel of each labeled point emphasize the heads in the image. The dynamic density map generator model is shown in Fig. 3.

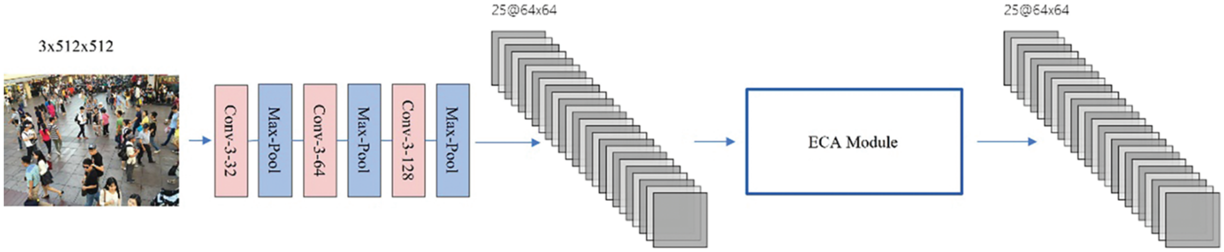


Figure 3: Dynamic density map generator model (Conv-kernel-channels)

The dynamic density map generator model takes a three-channel image with a resolution of 512 × 512 as input. Supposing the output channel of the last convolution layer is 25 ($k = 5$), it indicates that each ground-truth annotated point is represented by a 5 × 5 matrix. The ECA module is appended to help all the channels focus on the heads' features. The ECA module is shown in Fig. 4.

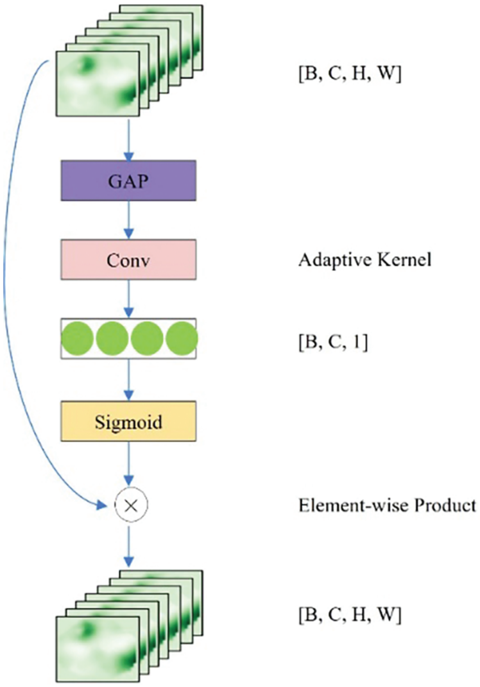


Figure 4: Efficient channel attention module

2.2 *Lightweight CSRNet*

There are many outstanding crowd-counting models in the academic field, such as SASNet, ADSCNet and SFANet. Most of these models take advantage of diversified techniques to improve performance, but deploying those models will be more difficult for embedded devices with limited computing resources. For example, when the computing resources of a model with massive parameters are required beyond the capacity of the embedded device or the device lacks Compute Unified Device Architecture (CUDA) cores, it is difficult to take advantage of the excellent model based on multi-column structures. Even worse, the device may be unable to maintain compatibility between the novel techniques and the hardware at hand. Therefore, with the tradeoff of performance, parameters quantity and complexity of the model, this paper adopts CSRNet as the crowd-counting backbone network structure, which is a single-column-structure-based network.

Since visual geometry group network (VGG)-16 [23] has a strong transfer learning ability, CSRNet selects the first ten network layers of the VGG-16 network as the image feature extraction network, which reduces the resolution of the input image to 1/8 of the original size. Keeping the resolution of the feature map unchanged, six dilated convolution layers with the convolution kernel size of 3 and the dilation rate of 2 (followed by rectified linear unit activation function) are used to expand the receptive field for feature extraction. Then we obtain the single-channel density map after a regression layer and sum the elements of the map as the crowd-counting result.

Although CSRNet shows outstanding performance with a relatively small amount of parameters, it still encountered difficulties in deployment on embedded devices. Therefore, this paper takes measures to further reduce the parameters of CSRNet by reducing the output channels of the network and introducing the CSP module. From our perspective, CSRNet can be divided into six parts before the last regression layer shown in Fig. 5. We directly reduce the output channels of the first five parts, and Table 1 indicates the output channels of the network before and after the compression. In addition, inspired by CSPNet, this paper applies the CSP module to the compressed CSRNet from part 2 to 5. Ahead of each first layer in the above five parts, feature maps are divided into two halves in the channel dimension. Then one half is taken as the input of one extra 1×1 convolution layer that outputs half of the original channels. Meanwhile, the original layers of compressed CSRNet handle the other half of the input and output channels. Next, feature maps are concatenated in the channel dimension for the input of the subsequent phase. Additionally, to avoid each part of the CSRNet network extracting the feature information from the same half of the channels and ignoring the other half, this paper reverses the order of two halves each concatenating, which helps introduce diverse feature information to the deep layers. The final structure of CSRNet is shown in Fig. 6.

2.3 Online Knowledge Transfer

After reducing model parameters in Section 2.2, to enhance the feature-extracting ability of lightweight network and compensating for the performance deterioration, this paper trains a lightweight crowd counting model with knowledge transfer. In this paper, with CSRNet used as the teacher model and the lightweight network generated in the previous section used as the student model, the SKT training framework is utilized to speed up the fitting speed of the student model to the dataset and enhance the learning ability of the student model to make up for the accuracy loss.

The original SKT framework adopts a two-stage training mode, in which the teacher model is trained regularly and the best one is involved in the coming phase of training the student model. To further reduce the workload, we optimize the teacher and student model simultaneously based on the SKT framework, which conducts the knowledge transfer while training the teacher network. During the training, the teacher model remains in the training phase. However, we define the training condition, especially for the student model, in which the student model should stop being optimized if its evaluation performance is superior to the teacher model. Otherwise, it will continue training. The detailed training procedure is shown in Algorithm 1.

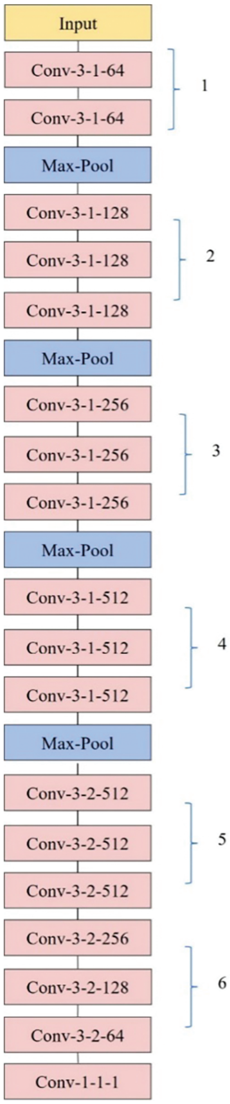


Figure 5: Six blocks of CSRNet

Table 1: Output channels of congested scene recognition network before and after channel compression

CSRNet structure	Before	After
1	64	32
2	128	64
3	256	128
4	512	256
5	512	256

(Continued)

Table 1: Continued

CSRNet structure	Before	After
6	256	256
	128	128
	64	64

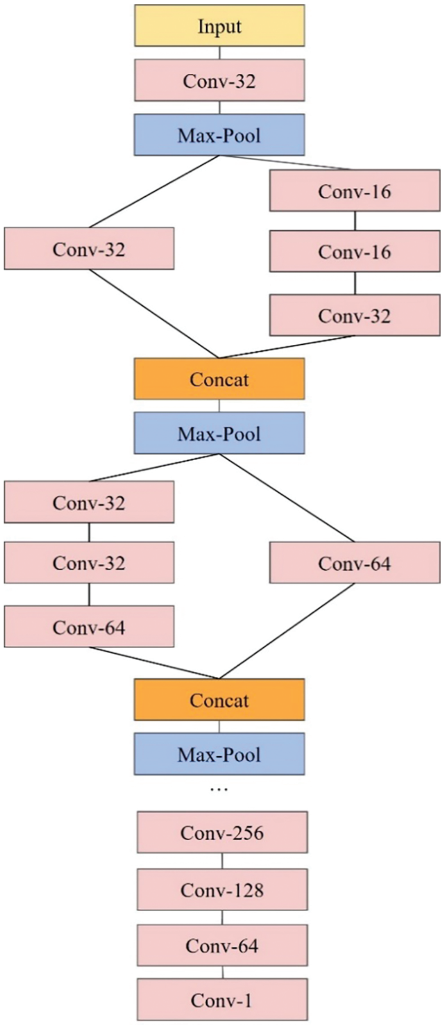


Figure 6: Final structure of CSRNet

Algorithm 1: Online structured knowledge transfer

```
Input: training images and corresponding annotated maps
Output: dynamic density map generator model, teacher model and student model
1. $StudentKeepTraining = True$ // flag to indicate whether to train student model
```

(Continued)

Algorithm 1: Continued

```
$MAE_{teacher\_best} = 1e6$, $MAE_{student\_best} = 1e6$ // mean absolute error (MAE) of teacher and student model
for $m = 1 \rightarrow M$ do // m for epoch
  for $i = 1, 2, \ldots, N$ do // I for the number of the image
     density map $P_i$ of teacher network
      density map $G_i$ of dynamic density map generator model
     update parameters of the teacher and the dynamic density map generator model
  end for
  $MAE_{teacher}$ for evaluation on teacher mode
     if $MAE_{teacher} < MAE_{teacher\_best}$ then
     $MAE_{teacher\_best} = MAE_{teacher}$
             current teacher and dynamic density map model as $Best\_Teacher$ and $Best\_Genetator$
     end if
     $StudentKeepTraining = True$
     if $MAE_{student\_best} < MAE_{teacher\_best}$ then
       $StudentKeepTraining = False$ // stop training student model
     end if
     if $StudentKeepTraining == True$ then
       for $i = 1, 2, \ldots, N$ do // $i$ for the number of the image
           density map $P_{teacher-i}$ of $Best\_Teacher$
           density map $G_i$ of $Best\_Generator$
           density map $G_{student-i}$ of student model
           update parameters of the student model
         end for
         $MAE_{student}$ for evaluation on student model
         if $MAE_{student} < MAE_{student\_best}$ then
           $MAE_{student\_best} = MAE_{student}$
         end if
         if $MAE_{student\_best} < MAE_{teacher\_best}$ then
           $StudentKeepTraining = False$ // stop training student model
       end if
    end if
end for
```

3 Performance Analysis

To evaluate the performance of our lightweight model, we conduct model training and testing on several crowd-counting datasets with security monitor views, such as Venice [9], Beijing-BRT [24] and ShanghaiTech Part B [3].

3.1 Dataset

The Venice dataset consists of 167 images, with 80 images for training and 87 images for testing, each resolution of which is fixed to 1280 × 720. The images feature Piazza San Marco as seen from various viewpoints on the second floor of the basilica, providing different monitor views to train crowd-counting models.

The Beijing-BRT dataset was collected at the Bus Rapid Transit (BRT) in Beijing, with 720 images for training and 560 for testing. The size of each image is 640 × 360 pixels, and the number of heads varies from 1 to 64. The images contain interference like shadows, glare and sunshine, and the time span is from morning to night, which is suitable for practical applications such as intelligent traffic.

The ShanghaiTech dataset contains 1198 images with a total of 330165 persons. It is comprised of two parts: Part A is composed of 482 images randomly selected from the Internet, including 300 images for training and 182 images for testing, each size of which is 868 × 589; Part B has 716 images taken from the streets of Shanghai city, with 400 images for training and 316 images for testing, each resolution of which is 1024 × 768. This paper only uses ShanghaiTech Part B to train and test models because images of Part B are mainly captured in a monitor view corresponding to our need.

3.2 Model Training

Model training is conducted on AI Studio platform provided by Baidu, which offers Intel (R) Xeon (R) Gold 6148 central processing unit (CPU) @ 2.40 GHz processors, 32 GB random-access memory (RAM) and Nvidia Tesla V100-SXM2 graphic cards. We train models based on the Paddle 2.2.2 framework and transform the Paddle model into the ONNX model with operator set 11.

During model training, training images are randomly cropped as half or 1/4 of the original resolution and horizontally flipped as the data augmentation. The Adam optimizer is utilized to optimize the networks. The learning rates of the teacher network, the dynamic density map generator model and the student network are 5×10^{-7}, 1×10^{-7} and 1×10^{-4}.

This paper optimizes the dynamic density map generator model, the teacher model and the student model with corresponding loss functions.

In terms of the dynamic density map generator model and teacher model, it takes the following loss function as the optimizing target:

$$L = \sum_{i}^{N} ||P_i - G_i||^2 + \alpha \left(1 - CS\left(P_i', G_i'\right)\right), \tag{2}$$

where N is the number of training images. P_i and G_i indicate the density maps generated by the crowd-counting teacher model and the dynamic generator model, respectively, while P_i' and G_i' represent the corresponding vector forms. α is a hyperparameter, and $CS\left(P_i', G_i'\right)$ indicates the cosine similarity between P_i' and G_i'.

The generator model is optimized only during the training of the teacher model. After each validation of the teacher model, the current generator model is considered as a better one if the validation performance of the teacher model outperforms its previous best result. Therefore, a better generator model is saved and used to generate ground-truth density maps for training the student model as “*Best_Genetator*” in the next phase. This procedure is also stated in Algorithm 1. Visualizations of ground-truth density maps are shown in Fig. 7. Compared to traditional methods, our generator model renders a density map that shows a consecutive trend indicating the degree of crowdedness in the scene. Compared to the generator without the ECA module, our generator model helps highlight the crowded area in the density map.

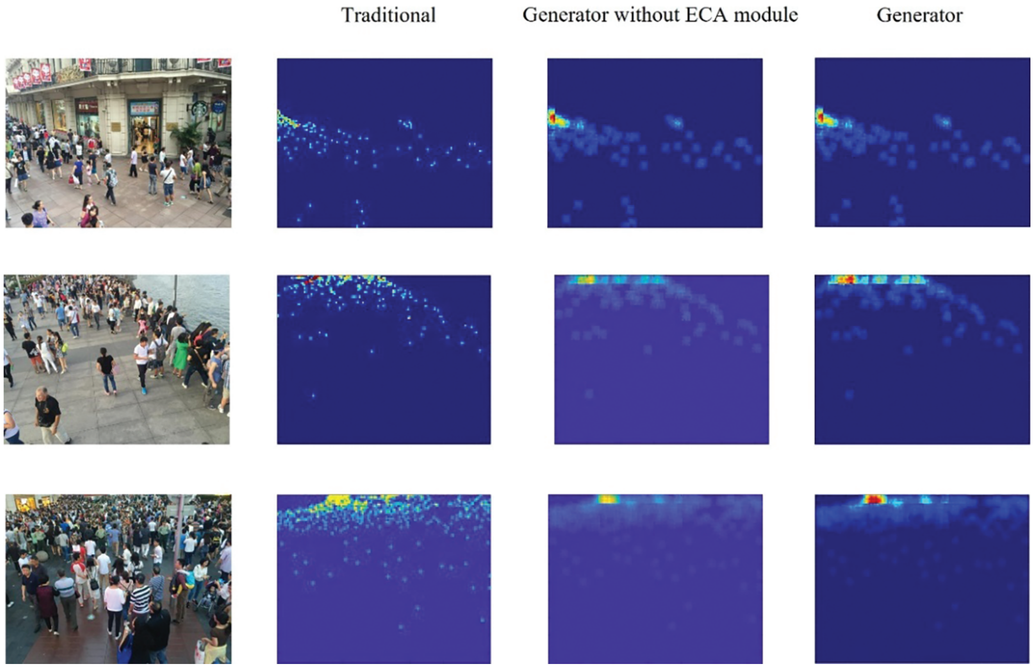


Figure 7: Visualizations of ground-truth density maps

In the student model, this paper adopts the loss function in the SKT framework as the optimizing target, i.e.,

$$L = \alpha_1 \cdot L_{Intra} + \alpha_2 \cdot L_{Inter} + \alpha_3 \cdot L_m, \tag{3}$$

$$L_m = \sum_i^N ||P_{i-student} - G_i||^2 + ||P_{i-student} - P_{i-teacher}||^2, \tag{4}$$

where α_1, α_2 and α_3 are hyperparameters. L_{Intra} is the intra-layer pattern transfer loss in the SKT framework. L_m is the sum of two Euclidean distances: 1) Euclidean distance between density maps of the student network and the static ground truth; 2) Euclidean distance between density maps of the teacher model and the student model. As shown in formula (4), we replace the first static ground-truth density maps with the density maps generated by our dynamic generator model.

3.3 Dataset Evaluation Performance

In the phase of testing models, MAE and root mean squared error (RMSE) are employed to evaluate the performance of the models. MAE and RMSE are defined as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^N |P_i - G_i|, \tag{5}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^N (P_i - G_i)^2}, \tag{6}$$

where N indicates the amount of testing images.

We evaluate the performance of our models based on the test set of Venice, Beijing-BRT and ShanghaiTech Part B. MAE, RMSE and model parameters are adopted to compare our model performance to that of some previous crowd counting researches such as MCNN [3], Swithcing CNN [25], CSRNet [4], dense feature extraction (DFE)-Crowd [26], attentive multi-stream (AMS)-CNN [27], context-aware network (CAN) [9], DecideNet [28], expanded context-aware network (ECAN) [9], geometric and physical constraints (GPC) [29], parasite-host network (PhNet) [30], hierarchical dense pyramid feature (HDPF) [31], hierarchical attention-based dense feature (HADF)-Crowd [32], fully convolutional network (FCN) [33], residual network (ResNet)-14 [34], deeply-recursive (DR)-ResNet [24], ResNet-DC [35], fusion of multi-layer features (FMLF) [36], modified segmentation network (M-SegNet) [6], M-SFANet [6], M-SFANet + M-SegNet [6], cross-hierarchy aggregation network (CHANet) [37], scale-adaptive convolutional neural network (SaCNN) [38], learning-to-rank (L2R) [39], multi-resolution crowd network (MRCNet) [40], MobileCount [19], Bayesian loss (BL) [41], distribution matching counting (DMCount) [42], deep structured scale integration network (DSSINet) [43], ADSCNet [10], SASNet [11], SANet [44], ShuffleCount [21] and cascaded multi-task learning (CMTL) [45]. Results are shown in Tables 2–5.

Table 2: Testing result on venice

Model	MAE	RMSE
MCNN	145.4	147.3
Switch-CNN	52.8	59.5
CSRNet	35.8	50.0
DFE-crowd	23.8	34.5
AMS-CNN	23.64	28.75
CAN	23.5	38.9
DecideNet	21.5	31.9
ECAN	20.5	29.9
GPC	18.2	26.6
PhNet	18.1	25.1
HDPF	16.3	23.9
HADF-crowd	14.1	20.1
Lightweight CSRNet (ours)	24.12	30.41
CSRNet	24.28	33.48

According to Tables 2–4, the performance on Venice shows that MAE and RMSE of our lightweight CSRNet drop 83.41% and 79.36% compared to MCNN, 54.32% and 48.90% compared to Switching-CNN, 32.63% and 39.18% compared to CSRNet, besides, the results are close to the DFE-Crowd, AMS-CNN and CAN. The performance on Beijing-BRT indicates that MAE and RMSE of our lightweight CSRNet drop 35.27% and 36.12% compared to MCNN, 16.67% and 11.94% compared to FCN, and the results are close to the ResNet-14 and other outstanding models. The performance on ShanghaiTech PartB reveals that MAE and RMSE of our lightweight CSRNet drop 58.7% and 56.2% compared to MCNN, 49.5% and 45.8% compared to Switching-CNN, 32.7% and 29.8% compared to SaCNN, 24.3% and 23.9% compared to L2R, and the results are close to CSRNet.

Table 3: Testing result on beijing-BRT

Model	MAE	RMSE
MCNN	2.24	3.35
FCN	1.74	2.43
ResNet-14	1.48	2.22
DR-ResNet	1.39	2.00
ResNet-DC	1.36	2.02
FMLF	1.34	2.02
M-SegNet	1.26	1.98
M-SFANet	1.16	1.90
M-SFANet + M-SegNet	1.15	1.81
CHANet	1.09	1.71
Lightweight CSRNet (ours)	1.45	2.14
CSRNet	1.42	2.12

Table 4: Testing result on ShanghaiTech Part B

Model	MAE	RMSE
MCNN	26.4	41.3
Switching CNN	21.6	33.4
SaCNN	16.2	25.8
L2R	14.4	23.8
CSRNet	10.6	16.0
MRCNet	10.3	18.4
MobileCount	9.0	15.4
MobileCount (∗1.25)	8.2	13.2
MobileCount (∗2)	8.1	12.7
CAN	7.8	12.2
BL	7.7	12.7
DM-count	7.4	11.8
DSSINet	6.85	10.34
ADSCNet	6.4	11.3
SASNet	6.35	9.9
Lightweight CSRNet (ours)	10.71	18.75
CSRNet	7.5	12.4

According to Table 5, when it comes to model parameters, our lightweight CSRNet owns 3.56 M parameters, compared to MobileCount (∗1.25), DSSINet, MobileCount (∗2), Switching-CNN, CSRNet, CAN, MRCNet, DMCount and BL, dropping by 34.92%, 59.82%, 73.42%, 76.44%, 78.11%, 80.34%, 82.47%, 83.45%. The model parameters of our lightweight CSRNet approach the amount of MobileCount, which is suitable for deploying on embedded devices.

Table 5: Model parameters

Model	Parameters (M)
DR-RESNET	0.028
MCNN	0.13
SANet	0.91
ShuffleCount	1.31
CMTL	2.46
MobileCount	3.40
MobileCount (∗1.25)	5.47
DSSINet	8.86
MobileCount (∗2)	13.39
Switching CNN	15.11
CSRNet	16.26
CAN	18.10
MRCNet	20.3
DMCount	21.50
BL	21.50
Lightweight CSRNet (ours)	3.56

According to Tables 2–4, the CSRNet of the original paper and the one we reproduce perform better in accuracy than our lightweight CSRNet. However, the results in Table 5 show that the original CSRNet model has more parameters leading to more memory cost in the application. Our lightweight CSRNet model has only almost 1/3 of the parameters of the original. Moreover, compared to other methods, the performance of our lightweight CSRNet model is close to the original CSRNet model, which maintains a balance between accuracy and parameters.

For evaluating the performance of our model under strong interference, one of the datasets used for training models (Beijing-BRT) contains data with interference like shadows, glare and sunshine. Judging by experiment results and visualizations in Fig. 8, the model can achieve corresponding performance under interference like shadows (the first column), glare (the second column) and fog (the third column).

To reveal the model performance in the actual scene, we conduct experiments on data collected by ourselves, which is not involved in the model training. Visualizations in Fig. 9 indicate that our models are still capable of analyzing the features even with interference, where there are many people dressed in the same color as the trees.

The above results demonstrate that our lightweight CSRNet retains acceptable performance, with a large number of model parameters reduced, indicating the feasibility of our method.

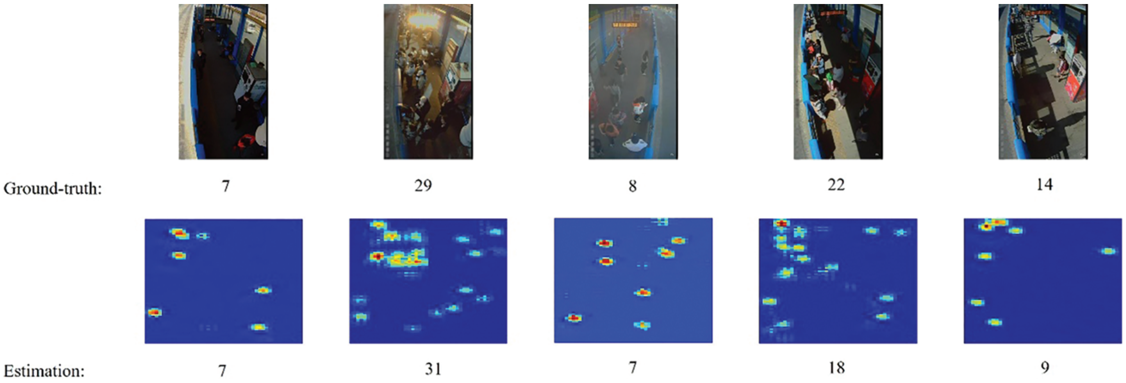


Figure 8: Experiment results and visualizations under interference

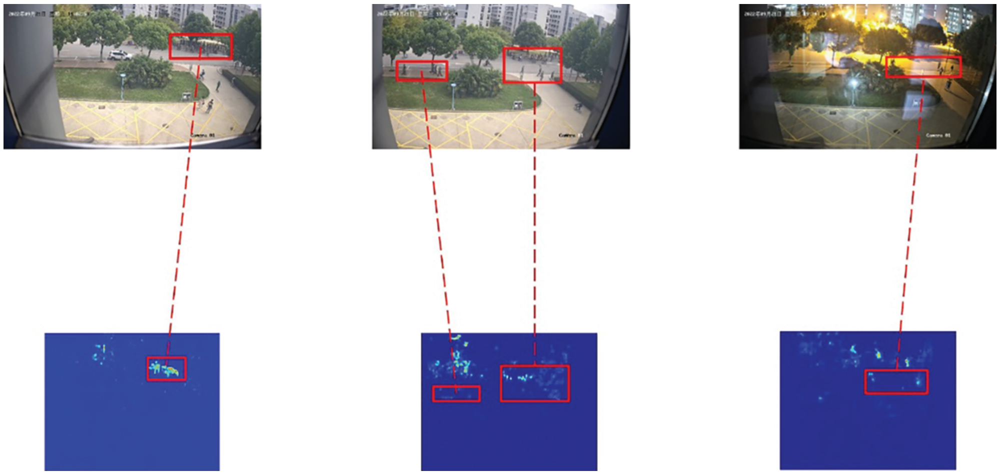


Figure 9: Visualizations of test results on unseen actual data under interference

4 Implementation of Neural Processing Unit Platform

Object detection-based solutions are often employed when encountering sparse scenes in crowd-counting tasks. For instance, Hu et al. [46] achieved crowd counting on the Nvidia Jetson Nano platform with YOLOv4 and Pazzaglia et al. [47] deployed MobileNetV3 [48] on the Rasberry Pi 4 platform to count pedestrians. We adopt the TB-RK3399ProD platform for crowd-counting model deployment. The TB-RK3399ProD platform is equipped with Neural Processing Unit (NPU). The NPU supports 8-bit or 16-bit inference and is compatible with multiple mainstream deep learning frameworks. After obtaining the corresponding ONNX model, it can be transformed into a quantized Rockchip Neural Network (RKNN) model for NPU to realize crowd counting on edge.

Apart from high-performance NPU, TB-RK3399ProD is also integrated with 2-channel MIPI camera serial interfaces, a raster graphic acceleration unit (RGA), a graphics processing unit (GPU) and a video processing unit (VPU). It also supports 10/100/1000 Mbps Ethernet data transmission. The platform is capable of capturing video streams and conducting preprocessing with RGA and GPU. After model inference on NPU, it transmits the video stream and related inference results to the users for analysis.

The crowd-counting procedure on the TB-RK3399ProD platform runs as follows. Supposing an IMX258 camera provides video streams, it supports capturing video streams with a resolution of 4192 × 3104 and a frame rate of 10 frames per second at best. Then the RGA unit is activated to finish the data copy, resolution adjustment and data format conversion from YUV to RGB within several milliseconds per action. Compared to processing on the CPU, RGA costs little consumption of CPU and RAM, for which data is flowing via direct rendering manager (DRM) or contiguous memory allocator (CMA). With resolution-adjusted RGB data, preprocessing can be conducted parallelly with OpenCL on the Mali-T860 GPU of the platform. Instead of consuming the CPU serially, it relieves the burden of CPU scheduling tasks. After RKNN model inference on NPU, the video stream processed is supposed to be encoded as H.264 or H.265 data stream for data transmission. It is often the common case that developers utilize multimedia frameworks like GStreamer or fast forward moving picture experts group (FFmpeg) to reach such a goal which requires high consumption of CPU and RAM. However, the platform offers the Media Processing Platform (MPP) library interface to control the video encoding and decoding hardware (VPU) to finish the encoding task. The VPU supports encoding 2-channel video streams with a resolution of 1920 × 1080 and a frame rate of 30 frames per second, which can meet the requirements of a practical application. Finally, Real Time Streaming Protocol (RTSP) data stream (with H.264 or H.265 data stream) and RKNN model inference results are transmitted through Ethernet to the users. The system procedure is shown in Fig. 10.

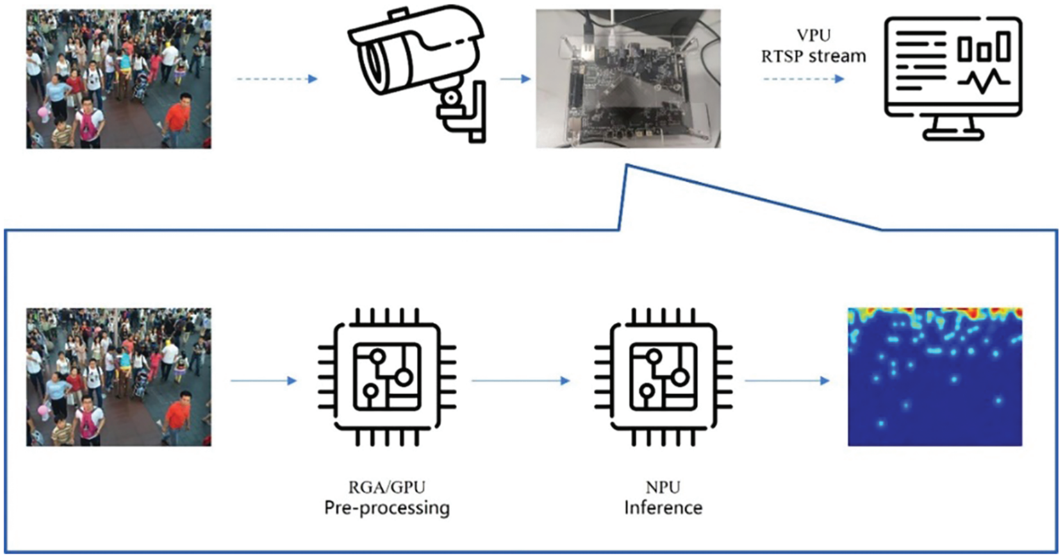


Figure 10: The procedure crowd counting system based on the TB-RK3399ProD platform

After deploying corresponding RKNN models on the TB-RK3399ProD platform, we evaluate the performance of the models by measuring MAE, RMSE and inference time. The results are shown in Table 6.

According to Table 6, the evaluation performance of RKNN models tested on the platform is slightly different from Paddle models tested on the AI Studio platform (server). On the Venice dataset, MAE and RMSE on the platform drop by 50.12% and 28.74% compared with those on the server. It takes 28 milliseconds to process a 3-channel image with a resolution of 640 × 360, which costs 23 milliseconds more than it does on the server. On the Beijing-BRT dataset, MAE and RMSE on the platform drop by 48.41% and 27.76% compared with those on the server. It takes nine milliseconds to process a 3-channel image with a resolution of 320 × 184, which costs four milliseconds more than it takes on the server. On the ShanghaiTech PartB dataset, MAE and RMSE on the platform grow by 0.18% and 0.64% compared with those on the server. It takes 20 milliseconds to process a 3-channel image with a resolution of 512 × 384, which costs 17 milliseconds more than it is required on the server. It is general that the performance variation is caused by the model quantization. According to Table 6, the performance on the platform is tolerable and acceptable. Visualization of some test results is shown in Fig. 11. What's more, the results of inference time indicate the system is competent to run inference in real-time, which verifies the feasibility of our solution.

Table 6: Evaluation results tested on the platform and server

Dataset	Image resolution	Model	MAE	RMSE	Inference time (ms)
Venice	640 ∗ 360 ∗ 3	RKNN	12.03	21.67	28
		Paddle	24.12	30.41	5
Beijing-BRT	320 ∗ 184 ∗ 3	RKNN	0.748	1.546	9
		Paddle	1.45	2.14	5
ShanghaiTech Part B	512 ∗ 384 ∗ 3	RKNN	10.73	18.87	20
		Paddle	10.71	18.75	3

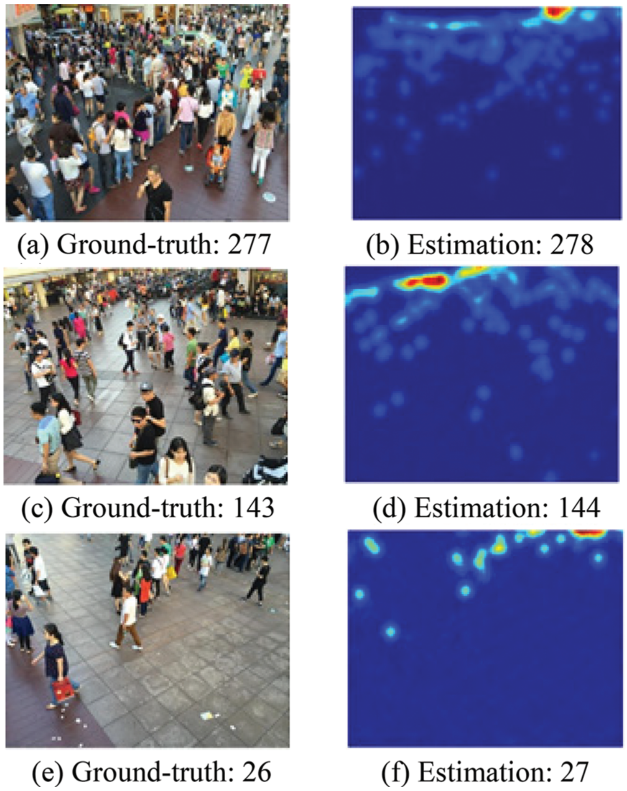

(a) Ground-truth: 277
(b) Estimation: 278
(c) Ground-truth: 143
(d) Estimation: 144
(e) Ground-truth: 26
(f) Estimation: 27

Figure 11: Visualization of test results on the platform

5 Conclusion

This paper proposes and implements a crowd-counting system for edge computing, which adopts CSRNet as the crowd-counting backbone network and introduces "cross stage partial" module to obtain a lightweight model for deployment on an embedded system. During the training, it takes a neural network to generate ground-truth density maps instead of traditional methods, and an efficient channel attention module is appended to enhance the feature-extracting ability of the generator model. Meanwhile, knowledge transfer based on the "structured knowledge transfer" framework is also employed to train teacher and student models simultaneously, which aims to compensate for the performance deterioration in an end-to-end manner. Experiment results show that our lightweight model has a competitive performance on several crowd-counting datasets, and our model maintains a balance between accuracy and inference time after deploying RKNN models on the TB-RK3399ProD platform. During the operation of the crowd-counting system, the platform can copy the data, scale the resolution and convert the data format of the real-time captured images based on the raster graphic accelerator unit. Meanwhile, the RGB data can be preprocessed by OpenCL parallel computing based on the platform graphics processing unit. After the preprocessed data is deduced on the neural network processing unit, the system can use the Media Processing Platform library interface to activate the video processing unit of the platform to encode the original data captured by the camera into H.264 or H.265 data stream. Moreover, the encapsulated Real Time Streaming Protocol data stream and the inference results are transmitted to the user terminal through the Ethernet data link. Based on the TB-RK3399ProD platform, this paper designs a lightweight deep-learning network model for the crowd-counting application and deploys it on the corresponding platform. With the rich hardware resources of the platform, the crowd-counting video surveillance system is realized, and a new embedded platform system scheme is provided for public safety pre-warning applications.

To make it possible to achieve crowd-counting application on embedded devices, this paper takes some measures to reduce a large number of parameters of the CSRNet for deployment and compensate for the accuracy loss. However, there exists a gap in accuracy between state-of-art models in the academic field and our lightweight CSRNet, even though it is successfully deployed, which inspires us to find out how to increase model accuracy while maintaining low quantities of model parameters. Furthermore, for the implementation of the crowd-counting system, the multimedia processing interface should be updated to be compatible with general multimedia frameworks like GStreamer or FFmpeg, which enhances the portability of our solution. In addition, our solution is designed as a crowd-counting camera product processing the single video data stream. At the same time, it is common that there are multiple surveillance cameras set up for the complex monitoring system. Therefore, it is also our future research to extend our work to enable our system to achieve multi-stream crowd counting, which requires few changes to the system architecture and makes it easier to develop other applications for the monitoring system.

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] A. Bochkovskiy, C. Wang and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed *et al.,* "SSD: Single shot multibox detector," in *Proc. of ECCV*, Amsterdam, NH, NLD, pp. 21–37, 2016.

[3] Y. Zhang, D. Zhou, S. Chen, S. Gao and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proc. of CVPR*, Las Vegas, NV, USA, pp. 589–597, 2016.

[4] Y. Li, X. Zhang and D. Chen, "CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes," in *Proc. of CVPR*, Salt Lake City, UT, USA, pp. 1091–1100, 2018.

[5] Z. Yan, Y. Yuan, W. Zuo, X. Tan, Y. Wang *et al.,* "Perspective-guided convolution networks for crowd counting," in *Proc. of ICCV*, Seoul, KOR, pp. 952–961, 2019.

[6] P. Thanasutives, K. Fukui, M. Numao and B. kijsirikul, "Encoder decoder based convolutional neural networks with multi-scale-aware modules for crowd counting," in *Proc. of ICPR*, Milan, ITA, pp. 2382–2389, 2021.

[7] L. Zhu, Z. Zhao, C. Lu, Y. Lin, Y. Peng *et al.,* "Dual path multi-scale fusion networks with attention for crowd counting," arXiv preprint, arXiv:1902.01115, 2019.

[8] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[9] W. Liu, M. Salzmann and P. Fua, "Context-aware crowd counting," in *Proc. of CVPR*, Long Beach, CA, USA, pp. 5099–5108, 2019.

[10] S. Bai, Z. He, Y. Qiao, H. Hu, W. Wu *et al.,* "Adaptive dilated network with self-correction supervision for counting," in *Proc. of CVPR*, Seattle, WA, USA, pp. 4594–4603, 2020.

[11] Q. Song, C. Wang, Y. Wang, Y. Tai, C. Wang *et al.,* "To choose or to fuse? Scale selection for crowd counting," in *Proc. of AAAI*, Vancouver, CAN, pp. 2576–2583, 2021.

[12] V. A. Sindagi and V. M. Patel, "Generating high-quality crowd density maps using contextual pyramid CNNs," in *Proc. of ICCV*, Venice, ITA, pp. 1861–1870, 2017.

[13] M. M. Oghaz, A. R. Khadka, V. Argyriou and P. Remagnino, "Content aware density map for crowd counting and density estimation," arXiv preprint arXiv:1906.07258, 2019.

[14] J. Wan and A. Chan. "Adaptive density map generation for crowd counting," in *Proc. of ICCV*, Seoul, KOR, pp. 1130–1139, 2019.

[15] J. Wan, Q. Wang and A. B. Chan, "Kernel-based density map generation for dense object counting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 3, pp. 1357–1370, 2020.

[16] G. Hinton, O. Vinyals and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint, arXiv:1503.02531, 2015.

[17] F. Zhang, X. Zhu and M. Ye, "Fast human pose estimation," in *Proc. of CVPR*, Long Beach, CA, USA, pp. 3517–3526, 2019.

[18] P. Wang, C. Gao, Y. Wang, H. Li and Y. Gao, "Mobilecount: An efficient encoderdecoder framework for real-time crowd counting," *Neurocomputing*, vol. 407, no. 24, pp. 292–299, 2020.

[19] L. Liu, J. Chen, H. Wu, T. Chen, G. Li *et al.,* "Efficient crowd counting via structured knowledge transfer," in *Proc. of MM*, Seattle, WA, USA, pp. 2645–2654, 2020.

[20] M. Jiang, J. Lin and Z. J. Wang, "Shufflecount: Task-specific knowledge distillation for crowd counting," in *Proc. of ICIP*, Anchorage, AK, USA, pp. 999–1003, 2021.

[21] C. Wang, H. M. Liao, Y. Wu, P. Chen, J. Hsieh *et al.,* "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proc. of CVPR*, Seattle, WA, USA, pp. 390–391, 2020.

[22] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo *et al.,* "ECA-net: Efficient channel attention for deep convolutional neural networks," in *Proc. of CVPR*, Seattle, WA, USA, pp. 11531–11539, 2020.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint, arXiv:1409.1556, 2014.

[24] X. Ding, Z. Lin, F. He, Y. Wang and Y. Huang, "A deeply-recursive convolutional network for crowd counting," in *Proc. of ICASSP*, Calgary, AB, CAN, pp. 1942–1946, 2018.

[25] D. B. Sam, S. Surya and R. V. Babu, "Switching convolutional neural network for crowd counting," in *Proc. of CVPR*, Honolulu, HI, USA, pp. 5744–5752, 2017.

[26] N. Ilyas, A. C. Najarro and K. Kim, "DFE-crowd: Dense feature extraction for single image crowd counting," *Symposium of the Korean Institute of Communications and Information Sciences*, vol. 2020, pp. 457–458, 2020.

[27] S. K. Tripathy and R. Srivastava, "AMS-CNN: Attentive multi-stream CNN for video based crowd counting," *International Journal of Multimedia Information Retrieval*, vol. 10, no. 4, pp. 239–254, 2021.

[28] J. Liu, C. Gao, D. Meng and A. G. Hauptmann, "Decidenet: Counting varying density crowds through attention guided detection and density estimation," in *Proc. of CVPR*, Salt Lake City, UT, USA, pp. 5197–5206, 2018.

[29] W. Liu, K. Lis, M. Salzmann and P. Fua, "Geometric and physical constraints for drone-based head plane crowd density estimation," in *Proc. of IROS*, Macau, CHN, pp. 244–249, 2019.

[30] S. Meng, J. Li, W. Guo, L. Ye and J. Jiang, "PhNet: Parasite-host network for video crowd counting," in *Proc. of ICPR*, Milan, ITA, pp. 1956–1963, 2021.

[31] N. Ilyas, Z. Ahmad, B. Lee and K. Kim, "An effective modular approach for crowd counting in an image using convolutional neural networks," *Scientific Reports*, vol. 12, no. 1, pp. 1–12, 2022.

[32] N. Ilyas, B. Lee and K. Kim. "Hadf-crowd: A hierarchical attention-based dense feature extraction network for single-image crowd counting," *Sensors*, vol. 21, no. 10, pp. 3483, 2021.

[33] M. Marsden, K. McGuinness, S. Little and N. E. O'Connor, "Fully convolutional crowd counting on highly congested scenes," arXiv preprint arXiv:1612.00220, 2016.

[34] K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition," in *Proc. of CVPR*, Las Vegas, NV, USA, pp. 770–778, 2016.

[35] J. Zhang, S. Chen, S. Tian, W. Gong, G. Cai *et al.,* "A crowd counting framework combining with crowd location," *Journal of Advanced Transportation*, vol. 2021, pp. 1–14, 2021.

[36] X. Ding, F. He, Z. Lin, Y. Wang, H. Guo *et al.,* "Crowd density estimation using fusion of multi-layer features," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 4776–4787, 2020.

[37] Q. Guo, X. Zeng, S. Hu, S. Phoummixay and Y. Ye, "Learning a deep network with cross-hierarchy aggregation for crowd counting," *Knowledge-Based Systems*, vol. 213, pp. 106691, 2021.

[38] L. Zhang, M. Shi and Q. Chen. "Crowd counting via scale-adaptive convolutional neural network," in *Proc. of WACV*, Lake Tahoe, NV, USA, pp. 1113–1121, 2018.

[39] X. Liu, J. V. D. Weijer and A. D. Bagdanov, "Leveraging unlabeled data for crowd counting by learning to rank," in *Proc. of CVPR*, Salt Lake City, UT, USA, pp. 7661–7669, 2018.

[40] R. Bahmanyar, E. Vig and P. Reinartz, "MRCNet: Crowd counting and density map estimation in aerial and ground imagery," arXiv preprint arXiv:1909.12743, 2019.

[41] Z. Ma, X. Wei, X. Hong and Y. Gong, "Bayesian loss for crowd count estimation with point supervision," in *Proc. of ICCV*, Seoul, KOR, pp. 6142–6151, 2019.

[42] B. Wang, H. Liu, D. Samaras and M. H. Nguyen, "Distribution matching for crowd counting," in *Proc. of NeurIPS*, Vancouver, CAN, pp. 1595–1607, 2020.

[43] L. Liu, Z. Qiu, G. Li, S. Liu, W. Ouyang *et al.,* "Crowd counting with deep structured scale integration network," in *Proc. of ICCV*, Seoul, KOR, pp. 1774–1783, 2019.

[44] X. Cao, Z. Wang, Y. Zhao and F. Su, "Scale aggregation network for accurate and efficient crowd counting," in *Proc. of ECCV*, Munich, BY, DEU, pp. 734–750, 2018.

[45] V. A. Sindagi and V. M. Patel. "CNN-based cascaded multi-task learning of high-level prior and density estimation for crowd counting," in *Proc. of AVSS*, Lecce, ITA, pp. 1–6, 2017.

[46] Z. Hu, W. Cao, F. Wu, Z. Zhang, C. Dong *et al.,* "A real-time UAV crowd counting system based on edge computing," in *Proc. of WCSP*, Changsha, CHN, pp. 1–5, 2021.

[47] G. Pazzaglia, M. Mameli, L. Rossi, M. Paolanti, A. Mancini *et al.,* "People counting on low cost embedded hardware during the sars-cov-2 pandemic," in *Proc. of ICPR*, Milan, ITA, pp. 521–533, 2021.

[48] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen *et al.,* "Searching for mobilenetv3," in *Proc. of ICCV*, Seoul, KOR, pp. 1314–1324, 2019.