An Auto-adaptive CNN for Crowd Counting in Monitor Image

Qiu Chang, Yonggang Qi, Wenli Zhou, Jun Liu

Beijing University of Posts and Telecommunications, Beijing 100876, China
qqstar15@163.com, qiyg@bupt.edu.cn, zwl@bupt.edu.cn, liujun@bupt.edu.cn

Abstract: The crowd computing in video surveillance has been a challenging task in the field of computer vision because of such problems as extreme overlap of objects, scale changes, case size, scene view and so on. Many works have focused on the issue of the scale variation, and have achieved great progress. To solve this issue, this paper presents an auto-adaptive deep convolutional neural network for crowd counting based on density: 1) a classifier is used to judge the density of image patches which are divided by image; 2) a regressor is used to predict the number of persons in high-density image patches, and a detector is used to predict the number of crowd in low-density image patches; and 3) we get the crowd counting of an image by adding up all the image patches of different level density.

To achieve further improvement from more and better data, we introduce PlayGround Crowd Dataset, a new set of person annotations on top of the playground dataset.

Keywords: Crowd counting; Density; Convolutional neural network (CNN); Faster R-CNN; Counting CNN

1 Introduction

With the construction of safe city, more and more public places are equipped with video surveillance, such as squares and supermarkets. Real-time monitoring of the number of people in video surveillance is becoming more and more important [1][2].

The purpose of population statistics is to count the number of people in the crowded scene. Almost all the statistical solutions of the crowd transform the problem into density estimation so that the image of the input crowd is mapped to the corresponding density map to represent the number of each pixel in the image [3]. Such issues as extreme overlap of objects, the scale variation, instance size, and scene perspective, are usually considered as the main issues for crowd counting, especially the scale variation issue.

Many methods of deep CNN [2][5][6] are proposed to solve the scale variation issue and have achieved good improvements. Namely, different sizes of convolutional kernels are applied to the input images to deal with different scaled humans, and the convolution maps from multiple-scale paths are fused to yield the final density estimation.

However, these works are density-specific, i.e., each counting model learned from a particular or a comparable crowd density can only be applied to the comparable density picture. In fact, the number of pedestrians in an image captured by surveillance video may be bustling, or as few as three or two. Even the two situations mentioned above appear in different places of one image in the meantime. When this happens, the single-trained network is no longer suitable.

In this paper, a deep learning model for density-based counting is proposed. On one hand, inspired by [6], we propose a patch-based dense estimation. First, cut one image into several small patches, such as 2, 4 or more patches, and estimate the density of each small patch separately. Then, to guarantee the accuracy of each different dense patch, feed the detection network with the patches with small density, feed the regression network with the patches with high density, and count the people number separately. Finally, the estimated numbers of the several pictures are added as the final result. The total process is shown in Figure 1.

The two contributions of this article. The purpose of this paper is to conduct accurate crowd statistic method by different deep learning models. So:

1. We propose an auto-adaptive deep CNN for crowd counting. The image is cut into several patches according to the picture quality, and each patch auto-adapts to different models.
2. We have collected a set of playground images from surveillance video. This dataset contains 356 labeled images, with a total of 6,337 people with centers of their heads labeled.

2 Related Work

Crowd counting has been added to the topic of computer vision for various reasons. There are many ways to count people in images. We can roughly divide those methods into two parts. One contains the traditional crowd counting methods, the other is mainly based on deep CNN [3].

Loy *et al*. [4] categorized the traditional crowd statistics methods based on the following categories: (1) Detection-based approaches, (e.g. head or body detection), (2) Regression-based approaches (e.g. HOG, SIFT, and Fourier Analysis), and (3) Density estimation-based approaches [3]. In fact, the performance of these weak representations based on local features is not as effective as deep representation.

978-1-5386-6067-6/18/$31.00 ©2018 IEEE

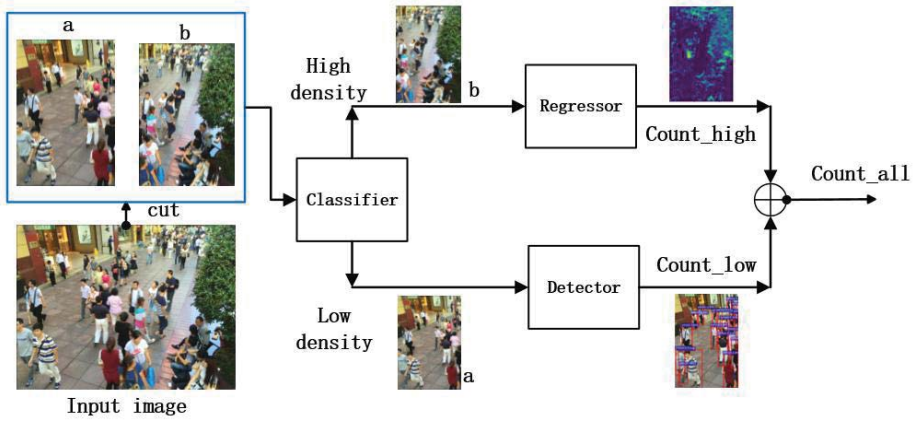

Figure 1 the architecture of the proposed network.

Recently, the solutions based on CNN become more and more popular. Onoro and Sastre [2] proposed a novel convolutional neural network solution, called Counting CNN (CCNN). The CCNN is essentially a regression model, and the task of the network is to learn how to map the appearance of the patches to its corresponding object density maps. They also proposed a scale-aware counting model, the Hydra CNN, a nonlinear regression model capable of learning multi-scale features. However, the counting performance of this model is very sensitive to the number of layers in the image pyramid. In [5], the huge changes in scale and perspective in crowd scenes are captured by shallow CNN columns with different receptive ranges. All methods have achieved very good results. Besides these regression-based models, there are other detector-based model studies [8][9]. [8] implemented an end-to-end method that accepts an image as input and then generates a set of object bounding boxes directly as output. However, when there are many pedestrians in the monitor screen, the problem of occlusion between pedestrians will be highlighted.

Both the methods use a custom network, trained separately for different scenarios. Therefore, the detection method is not well suited for high-density scenarios, nor as regression methods perform well in low-density scene.

For the images captured on video, the size of pedestrian density varies greatly, and the pedestrian density may be different in different places in one image. So we propose an auto-adaptive deep CNN for crowd counting. The image is cut into several patches according to the picture quality, and each patch auto-adapts to different models.

3 Our Approach

Facing the problem caused by different pedestrian density, we cut the monitoring images into several patches, each corresponding to the different pedestrian density and sent to different models for prediction. The sum of all the values is the final prediction, as shown in Figure 1.

3.1 Counting through object detecting

When pedestrians are scarce, we solve counting problem with the method of pedestrian detection. As one of the best object detection networks, Faster R-CNN [10] is chosen to realize this function in this paper.

The object detection system is basically made up of two modules. The first module is a deep fully convolution network that proposes the detected areas of potential pedestrians, and the second module is the Fast R-CNN [12] detector that uses the areas.

A deep convolution network is proposed, which is called RPN (Regional Proposal Network). It takes images and outputs a number of rectangular candidates, which may include pedestrians. The RPN can simultaneously predict the boundary and score of a pedestrian. The architecture adds on CNN two more convolutional layers--a box-regression layer (*reg*) and a box-classification layer (*cls*). Here k represents the maximum number of possible proposals for each sliding-window. The *reg* layer contains $4k$ outputs corresponding to the coordinates of k boxes, and the *cls* layer contains $2k$ outputs corresponding to the scores of estimated probability whether each proposal contains pedestrian. The k proposals, named anchors here, are parameterized relative to k reference boxes.

After RPN, Fast R-CNN is adopted to be the detection network. For an image, the loss function is defined as:

$$\mathrm{L}(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*). \quad (1)$$

Here, p_i represents the predicted probability of anchor i being a pedestrian. If the anchor is positive, the ground-truth label p_i^* will be 1, and if the anchor is negative , the p_i^* will be 0. Vector t_i represents the 4 parameterized coordinates of the predicted bounding box, so is t_i^* the ground-truth box associated with a positive anchor. For the classification loss L_{cls}, we use log loss over two classes (pedestrian vs. not pedestrian). The regression loss is $L_{reg}(t_i,\ t_i^*) = R(t_i - t_i^*)$ where R is the robust loss function (smooth L_1) defined in [12]. The term $p_i^* L_{reg}$ indicates that the regression loss is

only activated against the positive anchors ($p_i^* = 1$), otherwise disabled ($p_i^* = 0$).

3.2 Counting through density estimation

Another method of crowd counting in images is density estimation, whose purpose is to bypass the difficult task of learning single object instances. Therefore, the problem of counting turns to a problem of compute the continuous density function for the population in the image, and the integral in any image area is the number of people in the same area.

Firstly, a set of dot-annotated training images are marked, on which a dot represents a person. A dot image is shown in Figure 7 (a). For an image I, we define the ground truth density map D_I as:

$$D_I(p) = \sum_{\mu \in A_I} N(p; \mu, \sum), \tag{2}$$

where A_I is the 2D point set tagged by the image *I*. The evaluation of a normalized 2D Gaussian function is represented by $N(p; \mu, \sum)$. μ represents the mean value, $\sum$ represents isotropic covariance matrix, and the position for each pixel is expressed with p.

Through the density mapping D_I, we can directly integrate $D_I(p)$ and get the total count object of image N_I, as follows,

$$N_I = \sum_{p \in I} D_I(p) \tag{3}$$

The counting system then maps the learning to the nonlinear CNN, which converts the feature vectors at each pixel to the density value, thus obtaining the density function value of the pixel.

In the training process, the coefficients Ω of the nonlinear mapping is optimized so that the density function generated by this mapping can match the ground truth density as closely as possible. There are many kinds of CNN to estimate the density. We choose the Counting CNN (CCNN) in [2]. Then we get an object density map prediction $D_{pred}^{(P)}$,

$$D_{pred}^{(P)} = \mathrm{R}(\mathrm{P}|\Omega) \tag{4}$$

where the set of parameters of the CNN model is represented by Ω, and p, as an input, is an image patch.

3.3 Classification model

As mentioned above, using only one network does not work well for all cases. We need a classifier to decide which network model to use.

We cut one image into several small patches for two reasons.

Firstly, in different places in one image, the probability of pedestrian density is very different. So one image may not apply to just one model.

Secondly, as [11] said, Faster R-CNN has achieved very successful results in general object detection, but does not show competitive results on popular pedestrian detection datasets. One of the reason of such unsatisfactory performance is that the convolution feature mapping of Fast R-CNN classifier using to detect small objects is not a good solution. Common pedestrian detection scenarios, such as intelligent monitoring and automatic drive, are presented by pedestrians in relatively small sizes. If Faster R-CNN directly detects the whole image, the network may miss some small pedestrian targets. Cut one big image into some small patches can prevent learning features becoming "plain" to some extent.

Put the cut patches into two models shown in 3.1 and 3.2, and get their predictions respectively. Each small patch labels the tag of model whose estimation is the closet to its ground-truth. The classifier for model selection can be trained by these patches and their labels.

Table 1 Comparison of two datasets: Num shows the number of images; Max shows the maximal people number; Min shows the minimal people number; Ave shows the average people number; Total shows the total labeled people number; Shang_A is the dataset of Shanghaitech Part A, Shang_B is Shanghaitech Part B.

Dataset	PGCD	Shang_A	Shang_B
Resolution	2560*1440	Different	768*1024
Num	356	482	716
Max	86	3,139	578
Min	0	33	9
Ave	17.80	501.4	123.6
Total	6,337	241,677	88,488

4 Experiments

4.1 Evaluation function

Roughly speaking, the mean absolute error (MAE) indicates the accuracy of the estimates, which is defined as follows:

$$\mathrm{MAE} = \frac{1}{M} \sum_{i=1}^{M} |z_i - \hat{z}_i|. \tag{5}$$

Here, M represents the total number of test images, z_i represents the actual people number in the ith image, and $\hat{z}_i$ the estimated people number in the ith image.

4.2 PlayGround Crowd Dataset (PGCD)

We have collected a set of playground images from surveillance video. This dataset contains 356 annotated images, and a total of 6,337 people with centers of their heads are annotated. Table 1 gives the statics of PGCD and its comparison with other datasets.

The main feature of this dataset is that its vision is broad and pedestrians are scattered, as shown in Figure 2 (a).

Since the dataset is sparsely populated, we only use Faster R-CNN to detect persons, as shown in Figure 2 (b). In addition, the image size is too large, we divide it into 12 parts, i.e. 3*4. The experiment results of divided


)


b)

Figure 2 (a) an original image with the solution of 2560*1440 on PGCD; **(b)** the original image detected by Faster R-CNN directly.

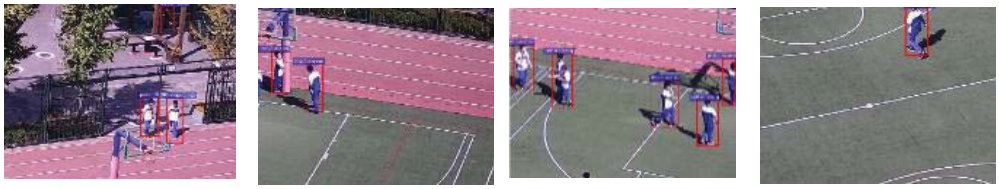

Figure 3 the results of patches detected by Faster R-CNN directly.


a)


b)


c)

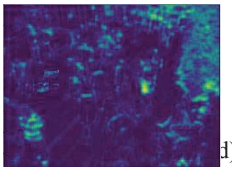
d)

Figure 4 (a) an original image on Part B; **(b)** the detect result of (a) with Faster R-CNN, total detect count is 27, the real value is 118; **(c)** the heat map of the dot image of (a); **(d)** the heat map of the predicted result as (a) experimented on CCNN .

Table 2 The objective evaluation of different methods on PGCD: Faster R-CNN represents the result of PGCD with original images experimented on Faster R-CNN directly; 12_ Faster R-CNN represents the result of PGCD with patches experimented on Faster R-CNN.

method	MAE
Faster R-CNN	15.28
12_Faster R-CNN	8.34

patches are shown in Figure 3 and Table 2.

4.3 Shanghaitech dataset

Shanghaitech dataset is introduced by [5]. The dataset contains 1198 annotated images, and 330165 people annotate the head center. The data set is made up of two parts: Part A has 482 images crawling randomly from the Internet, and Part B has 716 images from the bustling streets of Shanghai metropolis. Table 1 shows the statistical data of the Shanghai dataset and its comparison with other datasets. To fit the profile of video surveillance images better, we chose Part B as our experiment data. Here is an example of Part B shown in Figure 4 (a).

4.4 Experiments on Shanghaitech dataset

In our paper, we choose the Counting-CNN (CCNN) come up with [2] as our network for dense counting model, Faster R-CNN for detection counting model, and ResNet [13] for classifier.

Firstly, we compute the crowd count of each image on Part B with CCNN directly. Then we cut each image on Part B into 1*2, 2*2, 3*4 patches in order, and send them into CCNN for training and testing respectively. The results are shown in Table 3.

Since the performance is best when the image is cut into 1*2 patches, we choose this kind of cutting method to compute the crowd count of each cut image with Faster R-CNN. Then each cut image is tagged with the model whose estimate is more close to its real value. With the cut images and labels, we can train ResNet for classification. Finally, all the test dataset can be tested with the three trained models, and all the results are shown in table 4.

4.5 Analysis

From Table 2, we can find that for large-size, wide-view monitoring images, cutting methods can be very effective in improving detection quality. As can be seen from Figure 5, the detector performs better in a low-density scene and the regressor performs better in a high-density scene. Figure 5 proves that it is advisable to divide the images by density. Table 4 shows that the cutting of 2 patches by using Part B of Shanghaitech dataset has the best performance. The reason why this phenomenon occurs is probably because the density of most of images in Shanghaitech dataset is high, the overall characteristics of the crowd are obvious, so they are not suitable to cut too much, greatly different from PGCD. From Table 4, we can find that our method is the best in all comparisons, though the advantage is not obvious too much. The possible reason for not significantly improving performance of crowd counting is that our model is not an end-to-end model. The process of labeling and training the classifier may consume accuracy.

5 Conclusions and future work

In this paper, we have present an auto-adaptive deep

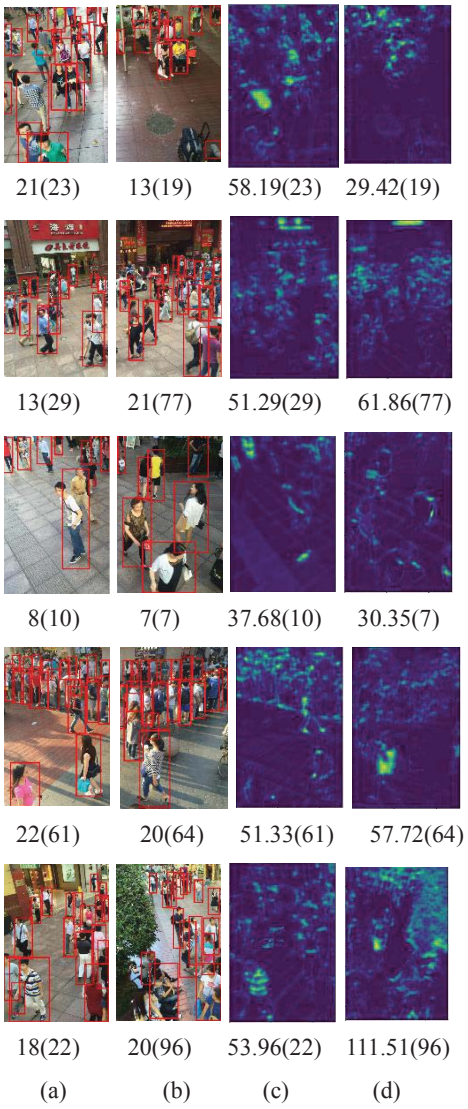

Figure 5 the data format is PredictValue (GT) **(a)** the detect result of left patch experimented on Faster R-CNN; **(b)** the detect result of right patch experimented on Faster R-CNN; **(c)** the heat map of left patch experimented on CCNN; **(d)** the heat map of right patch experimented on CCNN.

Table 3 Comparison of different cut forms of images on test set of Part B with CCNN: CCNN is the result of original images experimented on CCNN; 2_CCNN is the result of images cut into 1*2 patches experimented on CCNN; and so on.

Method	MAE
CCNN	57.07
2_CCNN	56.79
4_CCNN	58.18
12_CCNN	68.23

CNN for crowd counting based on crowd dense. The experiments show that the method we proposed can be used for crowd counting, especially in scenes where the level of the crowd density changes greatly. To improve the accuracy, we can also try other networks in the three models, such as SSD [15], SCNN [6], VGG [15], and so on. In addition, in order to enrich the dataset we have collected and annotated a new dataset named PGCD, which have proved that the cut patches are more suit for object detection than original images in wide-view scenes.

Table 4 Comparison of different methods on test set of Part B.

Method	MAE
CCNN	57.07
2_CCNN	56.79
2_Faster R-CNN	72.19
Our method	55.79

References

[1] Li T, Chang H, Wang M, et al. Crowded scene analysis: A survey[J]. IEEE transactions on circuits and systems for video technology, 2015, 25(3): 367-386.

[2] Oñoro-Rubio, Daniel, and R. J. López-Sastre. "Towards Perspective-Free Object Counting with Deep Learning." *European Conference on Computer Vision* Springer, Cham, 2016:615-629.

[3] Sindagi V A, Patel V M. A survey of recent advances in cnn-based single image crowd counting and density estimation[J]. Pattern Recognition Letters, 2017.

[4] Chen, C. L., Chen, K., Gong, S., & Xiang, T. (2013). *Crowd Counting and Profiling: Methodology and Evaluation*. *Modeling, Simulation and Visual Analysis of Crowds*. Springer New York.

[5] Zhang, Y., Zhou, D., Chen, S., Gao, S., & Ma, Y. (2016). Single-Image Crowd Counting via Multi-Column Convolutional Neural Network. *IEEE Conference on Computer Vision and Pattern Recognition* (pp.589-597). IEEE Computer Society.

[6] Sam D B, Surya S, Babu R V. Switching Convolutional Neural Network for Crowd Counting[J]. 2017.

[7] Zhang C, Li H, Wang X, et al. Cross-scene crowd counting via deep convolutional neural networks[C]. computer vision and pattern recognition, 2015: 833-841.

[8] Stewart, R., Andriluka, M., & Ng, A. Y. (2016). End-to-End People Detection in Crowded Scenes. *Computer Vision and Pattern Recognition*(pp.2325-2333). IEEE.

[9] Chen, S., Fern, A., & Todorovic, S. (2015). Person count localization in videos from noisy foreground and detections. *Computer Vision and Pattern Recognition* (pp.1364-1372). IEEE.

[10] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: towards real-time object detection with region proposal networks. *International Conference on Neural Information Processing Systems* (Vol.39, pp.91-99). MIT Press.

[11] Zhang, L., Lin, L., Liang, X., & He, K. (2016). Is Faster R-CNN Doing Well for Pedestrian Detection?. *European Conference on Computer Vision*(pp.443-457). Springer, Cham.

[12] Girshick, R. (2015). Fast r-cnn. *Computer Science*.

[13] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. 770-778.

[14] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., & Fu, C. Y., et al. (2016). SSD: Single Shot MultiBox Detector. *European Conference on Computer Vision* (pp.21-37). Springer, Cham.

[15] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *Computer Science*.