Counting Objects by Blockwise Classification

Liang Liu, Hao Lu, Haipeng Xiong, Ke Xian, Zhiguo Cao, and Chunhua Shen

***Abstract*— In this paper, we introduce the idea of blockwise classification to count objects. The current mainstream method for counting objects is to regress the density map or to regress the redundant count map via a deep convolutional neural network (CNN). However, these methods suffer from two critical issues: inaccurately generated regression targets and serious sample imbalances. First, the ground truth density map is generated by convolving the dot map using a Gaussian kernel. Because an inappropriate kernel can cover the background or uncover objects, this approach introduces a form of noise, and therefore results in ambiguities when training the networks. Second, inhomogeneously distributed objects often exist in images, which gives rise to a data collection bias. This leads to a long-tailed distribution of region counts, which is a typical characteristic that occurs with imbalanced samples; therefore, underestimations in high-density regions and overestimations in low-density regions are common. In this paper, we address these two issues within one framework—blockwise count level classification. The intuition behind this idea is that while it may not be possible to provide an exact count of pixels or patches, it is possible to provide a count of a region that falls within a certain interval with high confidence. Our method classifies the count levels of each block produced by nonlinearly quantizing the continuous counts, thus transforming the imbalance of sample patch counts into a class imbalance of count levels. Consequently, an information-entropy-inspired loss can be applied to alleviate this issue. Through ablative studies, we analyze the impact of imbalanced data, Gaussian kernel sizes, quantization errors, and the effectiveness of each module in our method. Without bells and whistles, our method outperforms or performs competitively with other state-of-the-art approaches on seven object-counting benchmarks, including four crowd-counting datasets from ShanghaiTech, WorldExpo'10, UCF-QNRF and UCF_CC_50, one vehicle-counting dataset (TRANCOS), one maize-tassel-counting dataset (MTC), and one challenging sonar fish-counting dataset that we constructed. The results suggest that our framework provides a strong and improved baseline for object counting.**

***Index Terms*— Object counting, convolutional neural networks, sample imbalance, count-level classification.**

Manuscript received May 6, 2019; revised July 15, 2019 and August 21, 2019; accepted September 15, 2019. Date of publication September 23, 2019; date of current version October 2, 2020. This work was supported by the Natural Science Foundation of China under Grant 61876211. *(Liang Liu and Hao Lu contributed equally to this work.)* This article was recommended by Associate Editor W. Li. *(Corresponding author: Zhiguo Cao.)*

L. Liu, H. Xiong, K. Xian, and Z. Cao are with the National Key Laboratory of Science and Technology on Multi-Spectral Information Processing, School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: wings@hust.edu.cn; haipengxiong@hust.edu.cn; kexian@hust.edu.cn; zgcao@hust.edu.cn).

H. Lu and C. Shen are with the School of Computer Science, The University of Adelaide, Adelaide, SA 5005, Australia (e-mail: hao.lu@adelaide.edu.au; chunhua.shen@adelaide.edu.au).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2019.2942970

I. Introduction

OBJECT counting is a key task in computer vision and has been widely applied to crowd monitoring [1], [2], cell statistics [3], and wildlife stock assessment [4]. Initially, counting tasks were conducted through object detection [5], [6]; however, later mainstream methods involve regressing a density map [7]. To date, object counting remains challenging because images have various appearances, scale changes, and inhomogeneous object distributions.

Object counting achieved great success when regression-based deep learning approaches were introduced [2]. Following this paradigm, the current counting methods mainly regress a density map [8], [9], [10] or redundant count map [11], [12], [13] via deep convolutional neural networks (CNNs). However, a performance gap still exists when using these methods. Through experiments, we observed that two critical issues significantly degrade the performance.

The first is inaccurately generated regression targets. The ground-truth density map is obtained by convolving the dot annotation map with a Gaussian kernel. In object-counting datasets, only dotted annotations are provided, but contain no explicit cues that specify the kernel size. Thus, the choice of Gaussian kernel is baseless: using different Gaussian kernels changes the ground-truth density map [11]. Moreover, as shown in Fig. 1(a), an inappropriate kernel covers only part of the object or may even cover the background, leading to pixel-level noise in the ground truth map. Second, data collection bias and inhomogeneously distributed objects in images lead to sample imbalance problems, especially for local patch counts. As shown in Fig. 1(b), the ground-truth patch counts follow a long-tailed distribution, which typically results in underestimations in the high-density regions and overestimations in the low-density regions [13], [14].

To address these challenges, we introduce the idea of blockwise classification for object counting. Intuitively, identifying the coarse count level for a region is much easier than determining an accurate count for a pixel or a patch because while finding an exact count is difficult, one can confidently predict that a count will be within a certain interval. Using this paradigm, by quantizing the count values into several bins according to a nonlinear quantization strategy, we transform the regression problem into a multiclass classification problem and ameliorate the quantitative differences between the count levels. Moreover, this approach transforms the problem of sample imbalance into the problem of class imbalance [15], [16], [17]; then, we can employ an information-entropy-inspired loss that applies only to discretized problems. In addition, because patch counting is robust to the Gaussian

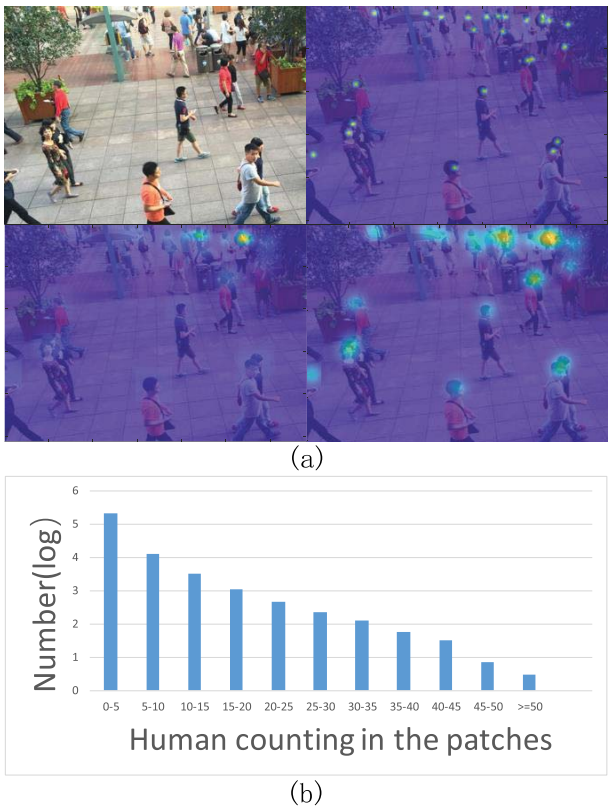


Fig. 1. The motivation of our method. (a) Because the Gaussian kernel lacks an object size annotation, it is not appropriate for objects of different sizes (upper and lower right) and introduces pixel-level noise when generating ground-truth density maps. The same logic applies to the adaptive kernel [1] (lower left). (b) The object distribution is imbalanced, especially for local patch counts (the example shows the statistics for 32×32 patches in the ShanghaiTech part_A dataset [1]) which all have the same order of magnitude after log calculation. We introduce the idea of blockwise classification to address these two issues within one framework.

kernel sizes [11], our method is not sensitive to noise produced by selecting an inappropriate Gaussian kernel size.

We empirically analyze the impact of imbalanced samples, Gaussian kernel sizes, quantization errors, and verify the effectiveness of each module in our method. Furthermore, we evaluate our method on four crowd-counting datasets (ShanghaiTech dataset [1], WorldExpo'10 [2], UCF-QNRF [18] and UCF_CC_50 dataset [19]), a vehicle dataset (TRANCOS dataset [20]), a maize-tassel-counting dataset (MTC dataset [12]), and a sonar fish-counting dataset that we constructed. The experimental results show that our method outperforms or performs competitively with other state-of-the-art counting methods. Overall, the main contributions of this paper are as follows:

- We propose the idea of counting by blockwise classification, forming a simple and robust baseline for object counting;
- We construct a new challenging sonar fish-counting dataset with manually labeled annotations;
- We report our framework's state-of-the-art performances on seven object-counting benchmarks.

The preliminary conference version of this paper appeared in [13]. Here, we extend [13] in following aspects. First, we inherit the patch-counting framework but replace the regression baseline with the classification baseline to further address sample imbalance. Second, we employ the information-entropy-based balance regularization from [13] to enhance the weights of minor samples according to the batchwise occurrence probability. In this paper, we adjust the weights based on the count probability levels, which we find to be more tractable and effective in practice. Third, we further conduct substantial experiments on other object counting benchmarks to demonstrate the generality of our method.

II. Related Work

The primary solutions for object counting can be classified into traditional methods and deep learning-based methods.

A. Traditional Methods

The traditional methods can be subdivided into two parts: detection-based approaches and regression-based approaches.

1) Detection-Based Approaches: The early solutions [5], [6] for object counting involved detecting objects by classifying handcrafted low-level features such as histogram oriented gradients (HOG) [21]. However, the detection approaches typically fail when there are small and dense targets because of variations in the objects' appearances and sizes. Moreover, it is difficult to label small targets with the bounding boxes that are typically employed in object detection tasks.

2) Regression-Based Approaches: To address the issues in detection-based approaches, regression-based approaches were proposed [22] that introduce the idea of global regression into the counting task. In a further evolution, the authors of [7] proposes regressing the density map, which is obtained by using kernel density estimation with Gaussian kernels. In addition, because a global regression loses spatial information and causes local regression to suffer from scalability issues, the work of [23] combines both global and local regression to estimate object counts. Moreover, the work of [19] employs multisource information to evaluate the counts and supply a confidence score by following the Markov random field approach to constrain the global consistency.

B. Deep Learning-Based Methods

The study by [2] may have been the first to introduce deep learning into object counting. This method regresses the counts and density map using deep CNNs. Adopting deep learning methods improved the accuracy significantly and quickly became the mainstream solution for counting. The authors of [24] proposed classifying the density level for entire images, and [11] and [12] employed redundant count maps to adapt to variations of the Gaussian kernel. Additionally, [18] attempted to estimate object locations, count objects and create a density map within one module. S-DCNet [25] employs spatial divide-and-conquer to transform open-set counting to closed-set one.

The main problem in object counting is to address the scale variations caused by perspective distortion and the different distances between objects and the camera. Multi-column CNN (MCNN) [1] is a popular solution for extracting multiscale

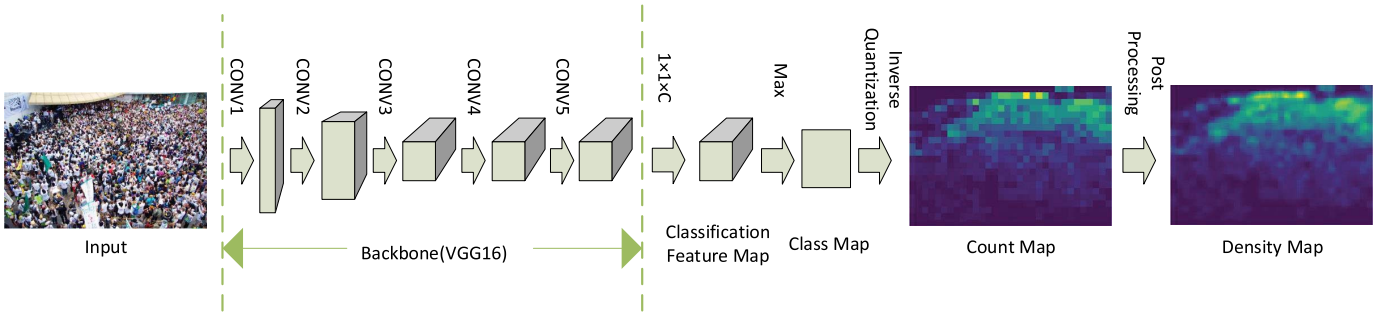


Fig. 2. An overview of our object-counting model. The input image is first sent through the 5 convolutional layer front-end, which has the same structure as VGG16. Then, a $1 \times 1 \times C$ convolution is added to form to the classification feature map. The max operator is used to choose the class map. After inverse quantization, the count map is obtained. A density map can be acquired after postprocessing.

features; it employs different filters sizes in each column to adapt to different scales and uses 1×1 filters to fuse the features. Based on MCNN, [26] trained a classifier to select only one column in MCNN to estimate the density map. [27] attempted to solve the issues of scaling variation and context simultaneously by replacing the 1×1 fusion filter in MCNN with a dilated convolution. In addition, [28] trained models at different scales and mixed their fully connected layers (FC layers) up. SCNET [29] employed residual fusion modules to extract a multiscale feature from a single column. DRSAN [30] employed a recurrent spatial-aware refinement module to perform an iterative optimization process to reduce the effects of perspective distortion.

In addition, because objects are inhomogeneously distributed in images, patch-level count [13] or image-level count [31] distributions are usually inhomogeneous. This imbalance issue is another aspect of studies focused on object counting. To adapt to imbalanced data, many works have employed classification as an auxiliary approach. The authors of [32] analyzed the effect of an inhomogeneous density map by decomposing the density map into several level maps and regressing each one. The author of [14] and [33] fuses the classification information for the regressors. Another work, [34], combines the detection output and the density map output to estimate the counts in the sparse and dense regions, respectively.

C. Limitations of the State-of-the-Art Approaches

Through the above summary of the previous works, we can see that the mainstream counting methods involve regressing the density map or regressing the redundant count map. The disadvantages of these methods can be summarized as follows:

- Because the available counting datasets commonly lack information about object size, error when selecting a Gaussian kernel is inevitable. An improper Gaussian kernel generates a noisy density map that tends to label background as targets or objects as background. This noisy ground truth introduces ambiguities to the model during the training stage.
- Objects tend to appear inhomogeneously in the image, which skews the distribution of the region-level and image-level counts. These distributions are long-tailed, and the majority of the patches are background or extremely sparse regions. Thus, they often lead to underestimations in high-density regions and overestimations in low-density regions [13], [14].

Fig. 3. The difference between a density map (left), a count map (middle) and a class map (right). Each point in the count map is the integration of a block in the density map. The class map is formed by quantizing the count map to the density level.

In this study, we address these two issues within a single framework. As mentioned above, we apply a blockwise classification baseline approach. Similar ideas of classification have also been proposed for other visual problems. For the depth prediction task, [35] proposed a solution that casts depth estimation as a classification task. [36] employed a similar soft weighted-sum strategy for robust depth prediction. Inspired by these analyses and proposed solutions, we introduce our method—counting objects by blockwise classification.

III. Counting by Blockwise Classification

In this section, we first present the details of our method, including how to obtain the count level from counts, the network architecture, the training loss, and the postprocessing operations. Then, we introduce the training details, including ground-truth generation, data augmentation, learning rate settings, and so on. Fig. 2 shows an overview of our method.

A. Count Level Generation

In this section, we introduce a method to quantize continuous patch counts into count levels. After quantification, we transform the network target into a count-level map and then a class map. The differences between the density map, count map, and class map are shown in Fig. 3.

To quantize the count map, the range from 0 to the maximum is divided into a series of bins following a certain division rule (we will discuss the division rule shortly). When a count lies between the upper and lower limit of a bin, it belongs to that bin category. The following formula describes the discretization process:

$$P(N_p \in C_i) = 1 \quad \text{if } L_i < N_p \leq U_i, \tag{1}$$

where N_p is the patch count, C_i is the i-th count-level class, L_i and U_i are the lower and upper limit of the i-th class, respectively, and $P(N_p \in C_i) = 1$ denotes that the probability of count N_p belonging to the i-th count level is 100%.

To recover the count map during prediction (inverse quantization), we set the median value of the bin to represent its class count. We use the following formula to define this process:

$$N_{C_i} = \frac{L_i + U_i}{2}, \tag{2}$$

where N_{C_i} is the corresponding count within the i-th count level.

When dividing the bins, one naive approach is to divide the bins uniformly. However, as shown in Fig. 1 (b), the patch counts in the dataset follow a long-tailed distribution; therefore, dividing the bins uniformly results in a huge class imbalance where sparse and background classes are much more prevalent than others. Thus, we divide the bins uniformly in the logarithmic space to narrow the steps in the sparse space and enlarge those in the dense space. In this case, we calculate the lower and upper limits of the i-th class L_i and U_i using the following formula:

$$L_i = \begin{cases} 0 & \text{if } i < 2 \\ e^{V+(i-2)\times q} & \text{otherwise,} \end{cases} \tag{3}$$

$$U_i = \begin{cases} 0 & \text{if } i = 0 \\ e^{V} & \text{if } i = 1 \\ e^{V+(i-1)\times q} & \text{otherwise,} \end{cases} \tag{4}$$

where q is the bin step in log space. As the above formulas show, we set the background to the dense class 0, whose lower and upper limits L_0 and U_0 equal 0. However, the minimum log space value cannot be set to 0 because there are a large number of patches whose counts are less than 1. Therefore, we set V as the start of the log space and divide bin $(0, e^V]$ into class 1 to make the bins continuous. In this study, we set V to -2; that is, the range of class 1 is $(0, 0.13]$.

B. Network Architecture

In this section, we introduce the architecture of our model. We employ VGG16 [37] as our backbone because it has a strong transfer learning capability [38]; then, we can fine-tune our model based on VGG16 pretrained on ImageNet [39]. VGG16 has five-stage convolution layers for encoding, and each stage includes two or three convolutions followed by ReLU. Each convolution stage is connected to a max-pooling layer. After max-pooling, the number of channel is doubled to compensate for the loss of the information. Two fully connected (FC) layers and a softmax layer are connected to the end of the 5-th max-pooling layer.

TABLE I

THE MODEL STRUCTURES OF THE COMPARED CLASSIFICATION AND REGRESSION

Baseline Structures	
Regression	Classification
Conv3-64	
Conv3-64	
Max-Pooling	
Conv3-128	
Conv3-128	
Max-Pooling	
Conv3-256	
Conv3-256	
Conv3-256	
Max-Pooling	
Conv3-512	
Conv3-512	
Conv3-512	
Max-Pooling	
Conv3-512	
Conv3-512	
Conv3-512	
Max-Pooling	
Conv1-1	Conv1-C
	Softmax

Our method utilizes only the encoding module in VGG16 to adapt to the resolution variations. To simplify further explanation, we define the following encoding calculation:

$$Y = f(X, m), \tag{5}$$

where X is the input image, m is the model parameter of the backbone, and Y is the encoded feature map. An 1×1 convolution $Conv1$, whose output channel number is set to the number of count levels, follows the backbone to output the block responses. Finally, we include a softmax layer to calculate the probability of each counted class. This process is defined in the following formula:

$$P = softmax(Conv1(Y)), \tag{6}$$

where P is the predicted blockwise count-level probability. Each point in P shows the count level of a 32×32 pixel block within the VGG16 backbone because—after the five max-pooling layers, each block whose size equals 32×32 pixels and stride equals 32 in the input image is downsampled to a 1×1 point. The network architecture of our method is depicted in Table I.

C. Loss Function

For classification, we typically employ cross-entropy loss as the training loss, which is defined as

$$\ell = -\frac{1}{N}\sum_{i=1}^{N} P(i) \log P(i), \tag{7}$$

where P is the probability map calculated by Eq. 6, i represents each point in P, and N is the number of blocks.

As Fig. 4 shows, after regrouping the counts by the log mapping, we alleviate the imbalance in the distribution. In a classification setting, the sample imbalance is transformed

to class imbalance. When training image-by-image (which means the batch size is 1 compared to training with minibatch images), class imbalance is an even more serious problem. Motivated by information-entropy-based balance regularization [13], we modify the loss weights to balance the different classes. To this end, we propose the loss with information entropy weight, which is defined as follows:

$$\ell = -\frac{1}{N}\sum_{i=1}^{N} IW(gt(i)) \times P(i)\log P(i), \tag{8}$$

where $gt(i)$ is the ground-truth class for point i in P, and IW is the information-entropy weight,

$$IW(GT) = -\log\frac{\sum_{j=1}^{N}(gt(j) == GT)}{N}. \tag{9}$$

The above formula adjusts the weight according to the probability of the class appearing in an image, which causes the less prevalent classes to have a greater contribution during training. Specifically, when all the patches have the same class (e.g., for a background image), IW is set to 1.

D. Postprocessing

After an image has been divided into patches, each patch count is estimated, and the summed count of each patch is usually not equal to the overall count of the original image [8]. This situation occurs because an object may become split during the prediction process. To address this issue, the work in [12] employs an overlapping sliding window to sample the patches. This approach improves model robustness because the overlapping sampling may include the entire object that was separated in an adjacent patch. Motivated by this work, we propose a similar intuitive solution for postprocessing. This postprocessing is equivalent to the postprocessing used in redundant counting [11] [12]. Instead of assessing an image only once, we send a series of offset images to the network. The image series $S(Image)$ is obtained as follows:

$$S(Image) = \{S_{ij} | S_{ij} = Image(x + X_{off\_i}, y + Y_{off\_j})\}, \tag{10}$$

where X_{off_i} and Y_{off_j} are the image offsets of x and y, respectively which are set to the values $Offset \in (-24, -16, -8, 0, 8, 16, 24)$ in this paper. To merge a series of estimated results, we average the patch count into each pixel and compute the merged density map C_m as follows:

$$C_m(x, y) = \frac{\sum_{i,j} C(x, y, i, j)}{N(x, y)}, \tag{11}$$

where $C(x, y, i, j)$ is the pixel count in the image with X_{off_i} and Y_{off_j}, and $N(x, y)$ is the number of times the pixel is computed in the series.

As discussed in Section III-B, we use five max-pooling layers in our model, which means that the output size is 32 times smaller than the input image, which results in checkerboard artifacts. The postprocessing can smooth such artifacts and slightly improve the performance by increasing the resolution of the final count map. Notice that, the postprocessing increases extra calculations. If the goal is not only to predict object counts but also generate the spatial distribution of objects, the postprocessing can be considered. If only object counts are needed, our classification method without postprocessing is sufficient.

E. Training Details

In this section, we describe the training details for our model, including ground truth generation, data augmentation, learning rate settings, and so on.

1) Ground Truth Generation: The goal of our method is to classify the count levels, which is achieved by processing the count map following subsection III-A. The count map is calculated by summing the pixel count in the patches, which is defined as follows:

$$N_p(i) = \sum_{x} D(x) \quad x \in p(i), \tag{12}$$

where N_p is the count map, $p(i)$ is the pixels in the i-th patch, x is the position of each pixel in $p(i)$, and $D(x)$ is the ground-truth density map.

2) Data Augmentation: Following [38], we crop the first four patches so that they contain the four quarters of the image; the other five patches are randomly cropped. All the patches are $1/4$ of the original image size. Moreover, we mirror these patches to double the training set. For the UCF_CC_50 dataset, we also crop other small patches from the $1/4$-sized patches because of the lack of samples in that dataset.

3) Training Details: The backbone convolutional layers are fine-tuned from the pretrained VGG16 model. The other layers are initialized following a Gaussian distribution with a standard deviation of 0.01 and a mean of 0. The learning rate of the pretrained layers is set to one-tenth of the learning rate applied to the Gaussian-initialized layers. In addition, the input image size must be divisible by the block size. Images whose size cannot meet this condition will be padded with zeros.

IV. Results and Discussions

In this section, we present the experimental results and analyses. Section IV-A shows the analyses the experimental results of our method and verifies the effectiveness of each module. Section IV-B compares the performance of our method with the current state-of-the-art methods. Section IV-C shows some failure cases and their analyses.

For comparisons, we adopt mean absolute error (MAE) and mean squared error (MSE) to evaluate the performances. These metrics are defined as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|C_p(i) - C_{gt}(i)\right|,$$
$$MSE = = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(C_p(i) - C_{gt}(i)\right)^2}, \tag{13}$$

where N is the number of images in the dataset, $C_p(i)$ is the predicted count of the i-th image, and $C_{gt}(i)$ is the corresponding ground-truth.

TABLE II

CONFIGURATION AND STATISTICS FOR EACH DATASET. ST MEANS SHANGHAITECH DATASET. '0', '0-5', ETC. INDICATE THE COUNT INTERVAL. 'A-PATCH' MEANS THE AVERAGE PATCH COUNT, 'M-PATCH' MEANS THE MAXIMUM PATCH COUNT, 'A-IMAGE' MEANS THE AVERAGE IMAGE COUNT, 'M-IMAGE' MEANS THE MAXIMUM IMAGE COUNT

Dataset	Kernel Size	Steps	0	0-5	5-10	10-15	15-20	20-25	25-30	≥30	A-Patch	M-Patch	A-Image	M-Patch
ST PartA [1]	6.0	0.05	81156	167263	11415	2964	989	387	233	201	1.29	58.71	501.4	3139
ST PartB [1]	Adaptive [1]	0.1	227712	78460	867	110	28	16	4	3	0.16	33.06	123.60	578
UCF_CC_50 [19]	Adaptive [1]	0.05	4202	8848	3576	2884	1514	820	566	1562	8.81	96.88	1279.48	4543
UCF-QNRF [18]	Adaptive [1]	0.1	7002632	1530112	17936	2954	623	157	77	27	0.12	40.57	815.4	12865
WorldExpo'10 [2]	Adaptive [1]	0.1	807431	273938	231	0	0	0	0	0	0.13	9.36	50.2	253
TRANCOS [20]	10.0	0.1	1090930	355034	0	0	0	0	0	0	0.03	2.99	33.61	95
MTC [12]	6.0	0.4	10152	15360	0	0	0	0	0	0	0.28	3.29	7.52	120
Sonar Fish	8.0	0.1	27346	14010	341	3	0	0	0	0	0.27	13.52	28.46	224

TABLE III

SAMPLE IMBALANCE ANALYSES ON THE SHANGHAITECH PART_A DATASET. 'BASELINE' MEANS THE LINEAR GROUPING CLASSIFICATION BASELINE, 'LOG' MEANS LOG GROUPING CLASSIFICATION, AND 'LOG+IW' MEANS LOG GROUPING WITH INFORMATION-ENTROPY WEIGHT

Class Step	Baseline MAE	Log MAE	Log+IW MAE
0.05	70.1	68.0	64.2
0.10	75.7	69.2	64.5
0.20	134.7	69.6	64.7
0.40	339.4	69.5	68.6

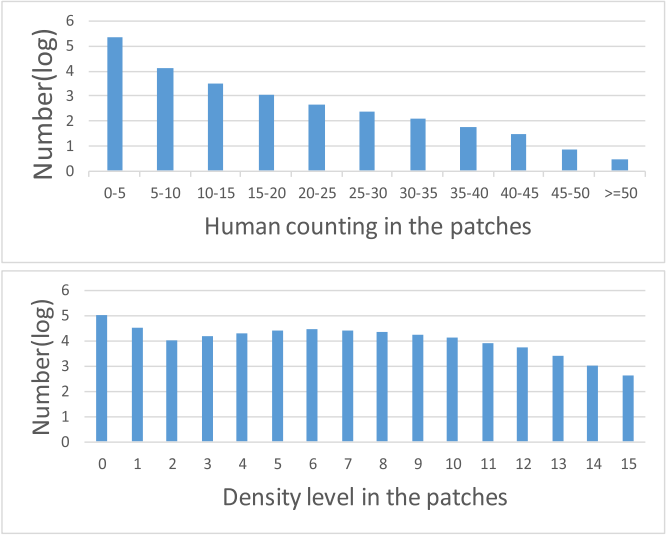


Fig. 4. The effect for log grouping in the imbalanced sample. These data are cropped from the Shanghai part_A dataset [1]. After the log grouping quantization, the imbalanced sample problem in patch counting is transformed into more balanced density level data (step equals 0.4).

A. Analyses of the Experiment for Blockwise Classification Counting

In this section, we analyze the experimental results of our method. All the experiments in this section are based on the ShanghaiTech part_A dataset [1] and the UCF-QNRF dataset [18], and no postprocessing is utilized except as noted.

1) Sample Imbalance and the Balance Process: The patch density level distribution quantized by linear grouping is basically identical to the distribution of the patch-counting values. In this distribution, extremely sparse and background patches comprise the majority of the samples, while the other patches are less prevalent. Thus, the sample imbalance is translated into a class imbalance, and the impact of sample imbalance can be analyzed by resorting to class imbalance techniques. In addition, we also verify the effectiveness of the balancing processes — log grouping and application of the information-entropy weights.

The results are shown in Table III, where *Baseline* represents the classification baseline with linear grouping, *Log* denotes the classification with nonlinear (log) grouping, and $Log + IW$ utilizes both log grouping and information-entropy weight for training. Specifically, we divide the linear spaces and the log space into the same number of bins, which are expressed using the log space steps for convenience.

Baseline shows that, as the linear step extends from (0.05 to 0.4), the class imbalance becomes more serious and the error increases sharply. This outcome illustrates that imbalanced distribution has a significant impact on the evaluated performance. The greater the imbalance in the distribution is, the larger the resulting error is. The results in column *Log* are all basically the same, and they are significantly decreased compared with the corresponding results in the *Baseline* column. This result occurs because—as shown in Fig. 4—log grouping reduces the distribution imbalance. Thus, these results verify the effectiveness of log grouping in adjusting the imbalanced data. In addition, the results in column $Log + IW$ are obviously better than the corresponding results in the *Log* column, which demonstrates that the information-entropy weight further adapts the model to the imbalanced data during the training stage. That is, this result verifies the effectiveness of applying the information-entropy weights.

2) Gaussian Kernel Adaptations: In this section, we analyze the Gaussian kernel adaptation for our method and the density-map-regression method. We compare CSRNET [38] with our method as representative of the density-map-regression baseline. We train the model with both the adaptive kernel [1] and with stable kernels. The results in Fig. 5 show that density-map regression is stable only within an interval of Gaussian parameters. When the kernel size is set to 1, its MAE sharply increases. In contrast, our method is more robust against Gaussian kernel variations, even when the kernel is set

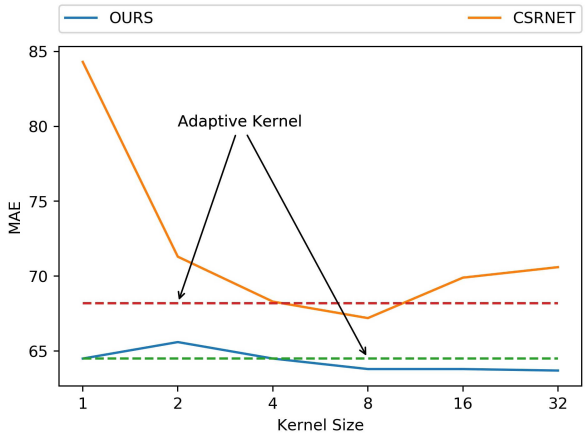


Fig. 5. Adaptation testing on the ShanghaiTech Part_A dataset of our method and density-map-regression baseline (CSRNET [38]) to various Gaussian kernels.

TABLE IV
EVALUATION OF QUANTIZATION ERROR ON THE SHANGHAITECH PART_A DATASET

Class Step	Quantization Error	Training Error
0.05	7.2	64.2
0.10	12.7	64.5
0.20	13.4	64.7
0.40	19.7	68.6

to 1. This experiment shows that our method is more adaptive to the Gaussian parameter.

3) Evaluation of the Quantization Error: In our method, we quantize the continuous values into bins and use the middle value of each bin as its counting value. This process inevitably introduces the quantization error. In this section, we analyze the effect of quantization error. First, in Table IV, we show the inherent quantization error (column *Quantization Error*) between quantized ground truth and unquantized ground truth, as well as the evaluation error of our method (column *Training Error*). The considerable difference between these two errors illustrates that quantization error is not the major contradiction of our method. Notice that we only quantize the ground truth to evaluate quantization error and to compute top-1 accuracy for regression in this subsection. In other parts of this paper, the quantized ground truth will not be used when reporting results. Second, to evaluate interval accuracy, Table V shows the block-level top-1 classification accuracy for patch count (top-1 in short). It also shows the image-level performance (MAE) by our classification method and by the local patch regression baseline. In addition, to evaluate block-level counting errors, Table VI shows the block-level MAE. The model structure of the regression baseline is shown in Table I and utilizes the ℓ_2 loss during training.

To calculate the TOP-1 accuracy for regression, we quantize the ground-truth and estimated patch counting according to each log step employed in the classification method. Similarly, the interval accuracy here is only used to compare our method with regression baseline. Other experiments will not employ

TABLE V
BLOCK-LEVEL TOP-1 ACCURACY AND IMAGE-LEVEL MAE FOR THE REGRESSION AND CLASSIFICATION METHOD ON THE SHANGHAITECH PART_A DATASET

Class Step	Classification		Regression	
	TOP-1	MAE	TOP-1	MAE
0.05	37.1%	64.2	16.2%	73.3
0.10	38.1%	64.5	17.5%	
0.20	40.5%	64.7	19.7%	
0.40	46.0%	68.6	24.4%	

TABLE VI
BLOCK-LEVEL MAE FOR THE REGRESSION AND CLASSIFICATION METHODS ON THE SHANGHAITECH PART_A DATASET

Class Step	0.05	0.10	0.20	0.40	Regression
Block-level MAE	0.41	0.41	0.41	0.42	0.46

TABLE VII
EVALUATION OF SOFT QUANTIZATION ON THE SHANGHAITECH PART_A DATASET

Class Step	0.05	0.10	0.20	0.40
Hard MAE	64.2	64.5	64.7	68.6
Soft MAE	64.4	64.9	64.3	69.6

this metric. These tables show that the classification method is more accurate for both block-level and image-level prediction compared with the regression method. In addition, because the top-1 accuracy is less than 50%, the main error of the classification method is inaccurate prediction. Although we try to deal with sample imbalance in object counting, there still exists other gaps in counting such as illumination variations and scale variations. These problems further restrict the performance improvement of the counting method. Overall, these experiments demonstrate that the main error in the classification method stems from incorrect prediction rather than from quantization error. Second, the classification method evaluates the blockwise density level more accurately, which is why it achieves better performance than does the regression method, despite the existence of quantization error.

In addition, we evaluate the effectiveness of the soft-weighted sum inference strategy [40]. The results shown in Table VII demonstrate that the soft-weighted sum inference strategy is not useful in improving the performance. Thus, we do not employ the soft-weighted sum inference strategy in the subsequent experiments.

4) Grouping Steps Sensitivity: We evaluate the step sensibility by employing 8 different steps in the log space. Fig. 6. demonstrates that our method is not sensitive to bin step changes. As the steps increase, the performance decreases slightly because of the increase in quantization error.

5) Block Size Sensitivity: We evaluate the block size sensibility by employing three different block sizes. We build the model whose block size is 16 by removing the 5-th max pooling in the VGG16 backbone and the model whose block size is 8 by removing the 5-th and 4-th max pooling layers, while

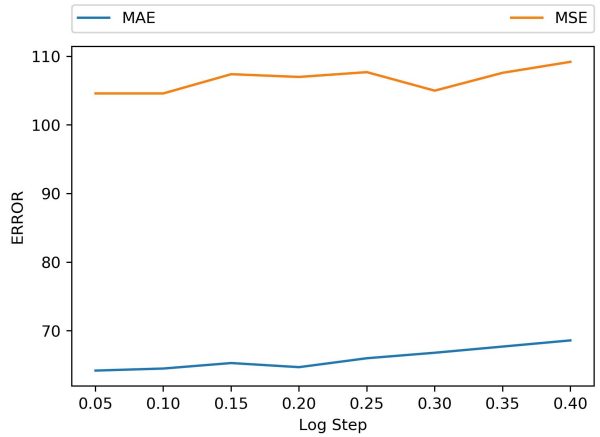


Fig. 6. Sensibility testing for the steps in the log space on the ShanghaiTech Part_A dataset.

TABLE VIII
SENSIBILITY OF BLOCK SIZES ON THE SHANGHAITECH PART_A DATASET

Size	MAE	MSE
8	97.6	138.6
16	64.8	107.6
32	64.5	104.6
64	64.8	101.1
128	72.3	111.9

TABLE IX
BACKBONE ADAPTABILITY, MODEL CAPACITY AND GFLOPS ON THE UCF-QNRF DATASET. GFLOPS ARE EVALUATED ON THE $224 \times 224 \times 3$ INPUT

Backbone	MAE	MSE	#Param.	GFLOPs
VGG16 [37]	140.6	218.7	1.47×10^7	200.3
MobileNetv2 [41]	135.9	206.9	2.93×10^6	4.6
ResNet101 [42]	127.3	199.3	4.36×10^7	102.7
DenseNet201 [43]	118.1	191.9	1.91×10^7	57.5

64-block-size model and 128-block-size model is obtained by adding an average pooling layer after the 5-th max pooling with appropriate strides. The results shown in Table VIII demonstrate that our method is not sensitive to block-size changes within a wide range.

6) Adaptation to Different Backbones: Here we verify the adaptation of our method to different backbones. We test our method with four different backbone in the UCF-QNRF dataset [18] – VGG16 [37], MobileNetv2 [41], ResNet101 [42] and DenseNet201 [43]. Results shown in Table IX demonstrate that our method achieves better counting performance by leveraging further structured backbones. We also show model capacity and GFLOPs in Table IX.

7) Postprocessing: In this section, we evaluate the effect of postprocessing. The mean and standard deviation (calculated from 5 independent runs) of MAE and MSE with/without postprocessing are reported in Table X. We observe that the postprocessing can improve the performance slightly and also stabilize the prediction. The processed density examples

TABLE X
ABLATION STUDY OF THE POSTPROCESSING ON THE SHANGHAITECH PART_A DATASET. THE MEAN AND STANDARD DEVIATION OVER 5 INDEPENDENT RUNS FOR MAE AND MSE ARE PRESENTED

Step	Module	MAE	MSE
0.05	W/O P	63.9±0.2	103.8±0.7
	W/ P	63.5±0.4	103.5±0.9
0.10	W/O P	65.4±0.6	107.6±1.7
	W/ P	64.1±0.4	105.7±0.7
0.20	W/O P	64.7±0.8	107.0±0.6
	W/ P	63.8±0.7	105.4±0.3
0.40	W/O P	69.8±0.9	110.9±1.1
	W/ P	69.1±0.9	109.5±0.9

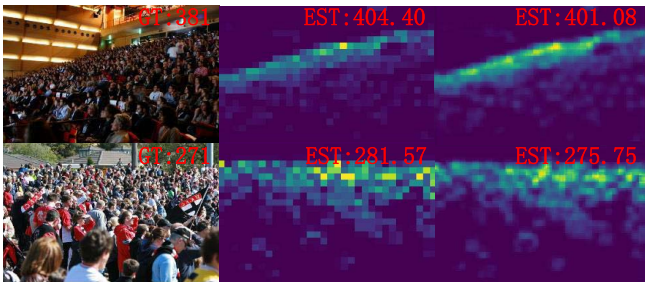


Fig. 7. The testing image samples in ShanghaiTech Part_A (column 1), the estimated count map (column 2) and the results after postprocessing (column 3).

TABLE XI
ALGORITHM COMPLEXITY ANALYSIS

Module	W/O P	W/ P
FLOPs (224×224)	1.53×10^{10}	2.00×10^{11}
FLOPs (640×480)	9.40×10^{10}	1.22×10^{12}
Model Capacity (VGG16)	1.47×10^7	

shown in Fig. 7 demonstrate that postprocessing can reduce the checkerboard artifacts and increase the resolution of the estimated result. All the following experiments employ the postprocessing module. Moreover, in Table XI we show the model capacity (with VGG16 backbone) and FLOPs for 224×224 and 640×480 images with/without postprocessing.

B. Comparisons With State-of-the-Art Methods

We compare our method with the state-of-the-art approaches on seven datasets—four crowd datasets, one vehicle dataset, one maize-tassel-counting dataset, and a sonar fish-counting dataset that we constructed. The model configurations and statistics for each dataset are shown in Table II.

1) ShanghaiTech Dataset: The ShanghaiTech crowd-counting dataset was proposed by [1]. This dataset contains 1, 198 images with 330, 165 dotting labels. Part_A and Part_B are divided into training and testing sets. Part_A has 482 images with various resolutions; 300 of these are included in the training set, and the remaining 182 images form the testing set. The minimum count is 33; the maximum count is 3, 139; and the average count is 501.4. Part_B has 716 images; 400 for training and 316 for testing. Part_B is sparser than Part_A;

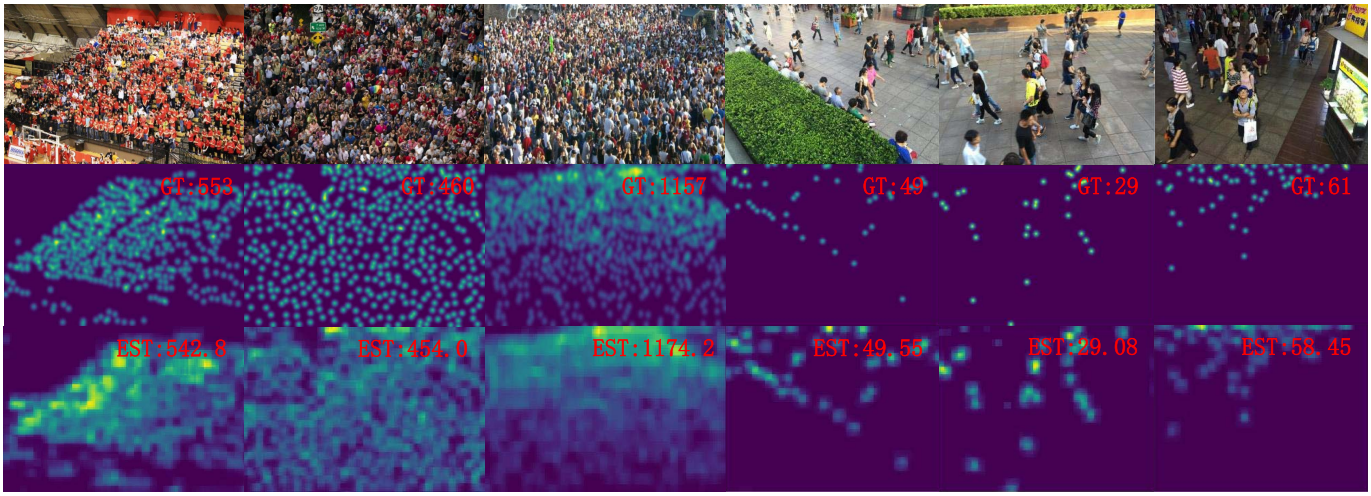


Fig. 8. Qualitative results on the ShanghaiTech [1] dataset. Row 1 shows the testing images, row 2 the corresponding ground truths, and row 3 our estimated results. Columns 1–3 show results from Part_A and columns 4–6 Part_B.

TABLE XII
ESTIMATION ERRORS ON THE SHANGHAITECH DATASET

	Part_A		Part_B	
Method	MAE	MSE	MAE	MSE
MCNN [1]	110.2	173.2	26.4	41.3
CP-CNN [14]	73.6	106.4	20.1	30.1
ACSCP [8]	75.7	102.7	17.2	27.4
L2R-Q [44]	72.0	106.6	14.4	23.8
L2R-K [44]	73.6	112.0	13.7	21.4
SCNet [29]	71.9	117.9	9.3	**14.4**
DRSAN [30]	69.3	**96.4**	11.1	18.2
ic-CNN [31]	68.5	116.2	10.7	16.0
CSRNET [38]	68.2	115.0	10.6	16.0
Ours	**62.8**	102.0	**8.6**	16.4

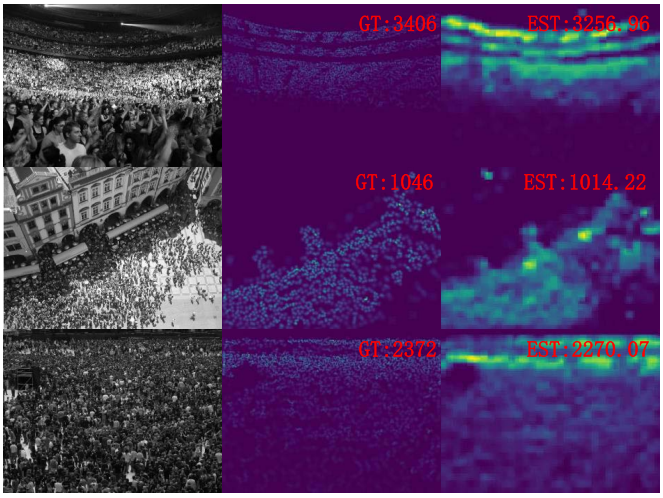


Fig. 9. Qualitative results on the UCF_CC_50 [19] dataset. Row 1 shows the testing images, row 2 the corresponding ground truths, and row 3 our estimated results.

the counts range between 9 and 578; and the average count is 123.6. The images have a fixed resolution of 1024×768.

We compare our method with the following 8 baselines. MCNN [1] leverages different size filters in each column to adapt to scale variations. CP-CNN [14] incorporates both local and global context with the classification information. ACSCP [8] employs a GAN to generate the density map and uses a regularizer to constrain the global consistency. Reference [44] introduces self-supervised learning into the counting task under different data retrieval strategies, in which L2R-Q means Query-by-example and L2R-K means Query-by-keyword. SCNET [29] employs residual fusion modules to extract a multiscale feature from a single-column. DRSAN [30] iteratively adopts a Recurrent Spatial-aware Refinement module to address the effects of changing perspective. CSRNET [38] replaces the FC layers of VGG16 with dilated convolutional layers. IC-CNN(two-stage) [31] leverages two columns to iteratively estimate the high resolution density map. The results shown in Table XII demonstrate that our method achieves the lowest MAE (e.g., the highest accuracy) on both Part_A and Part_B. Compared with state-of-the-art solutions, the MAE of our method is a decrease of 6% on Part_A and a decrease of 18.9% on Part_B, and our method achieves comparable MSE scores. We think the reason why other methods outperform ours in MSE scores is that, these methods employ MSE loss for optimization directly, while our method uses the cross-entropy loss instead. Some inferred examples are shown in Fig. 8.

2) UCF_CC_50 Dataset: UCF_CC_50 dataset [19], which was collected from web images, is a challenging and crowded dataset despite containing only 50 images. It is a high-density dataset in which the object counts range from 94 to 4, 543. The average count is 1279.5, and the dataset includes a total of 63, 705 labels. To evaluate the methods, following [19], the images are randomly divided into sets of 10, and fivefold cross-validation is employed. We evaluate our method and compare it with the same 8 methods described in Section IV-B.1. The results are shown in Table XIII and some estimation examples are shown in Fig. 9. The results

TABLE XIII
ESTIMATION ERRORS ON THE UCF_CC_50 DATASET

Method	MAE	MSE
MCNN [1]	377.6	509.1
CPCNN [14]	295.8	320.9
ACSCP [8]	291.0	404.6
SCNet [29]	280.5	332.8
L2R-Q [44]	291.5	397.6
L2R-K [44]	279.6	388.9
CSRNET [38]	266.1	397.5
ic-CNN [31]	260.9	365.5
DRSAN [30]	**219.2**	**250.2**
Ours	239.6	322.2

TABLE XIV
ESTIMATION ERRORS ON THE UCF-QNRF DATASET. METHODS WITH '*' REGRESS COUNTS WITHOUT COMPUTING DENSITY MAPS

Method	MAE	MSE
Idrees *et al.* [19]*	315	508
MCNN [1]	277	426
Encoder-Decoder [45]	270	478
CMTL [33]	252	514
SwitchCNN [26]	228	445
Resnet101 [42]*	190	277
Densenet201 [43]*	163	226
Idrees *et al.* [18].	132	**191**
Ours-VGG16	141	219
Ours-Densenet201	**118**	192

demonstrate that our method achieves parity with the other state-of-the-art approaches.

3) UCF-QNRF Dataset: The UCF-QNRF Dataset [18] is a challenging crowd counting dataset collected from Flickr, Web Search, and the Hajj footage. This dataset has 1,535 JPEG images with 1,251,642 head annotations. Its minimum count is 49 while the maximum count is 12865. High-resolution images are provided whose average resolution is 2013×2902. We report the results in Table XIV. Results show that our method (with VGG16 backbone) is comparable against state-of-the-art methods. In addition, as shown in Table IX, we have better performance by replacing the VGG16 backbone with further structured backbone. Especially, when we employ Densenet201 backbone as Idrees *et al.*, [18], our method reports the start-of-the-art performance. Some qualitative results are shown in Fig. 10.

4) WorldExpo'10 Dataset: The WorldExpo'10 dataset [2] includes 1132 annotated video sequences captured by 108 surveillance cameras from Shanghai 2010 WorldExpo. In this dataset, a total of 199,923 pedestrians are labeled within 3980 frames. 1127 video sequences out of 103 scenes are treated as training and validation sets, while other 5 video sequences from 5 different scenes are set as the testing set. We evaluate our method and compare it with 7 baselines. The quantitative results are shown in Table XV, and some qualitative examples

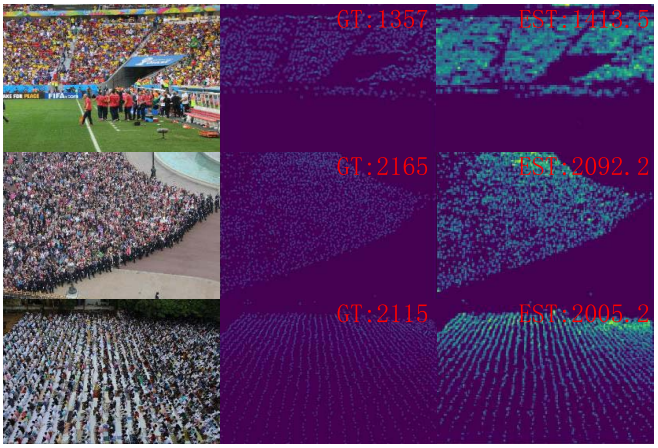


Fig. 10. Qualitative results on the UCF-QNRF [18] Dataset. Row 1 shows the testing images, row 2 the corresponding ground truths, and row 3 our estimated results.

TABLE XV
ESTIMATION ERRORS ON THE WORLDEXPO'2010 DATASET

Method	Scene1	Scene2	Scene3	Scene4	Scene5	Average
Zhang et al. [2]	9.8	14.1	14.3	22.2	3.7	12.9
CPCNN [14]	2.9	14.7	10.5	10.4	5.8	8.9
ic-CNN [31]	17.0	12.3	9.2	**8.1**	4.7	10.3
D-ConvNet-v1 [46]	**1.9**	12.1	20.7	8.3	2.6	9.1
DRSAN [30]	2.6	11.8	10.3	10.4	3.7	7.8
CSRNet [38]	2.9	11.5	8.6	16.6	3.4	8.6
ACSCP [8]	2.8	14.1	9.6	**8.1**	2.9	**7.5**
Ours	2.1	**11.4**	**8.1**	16.8	**2.5**	8.2

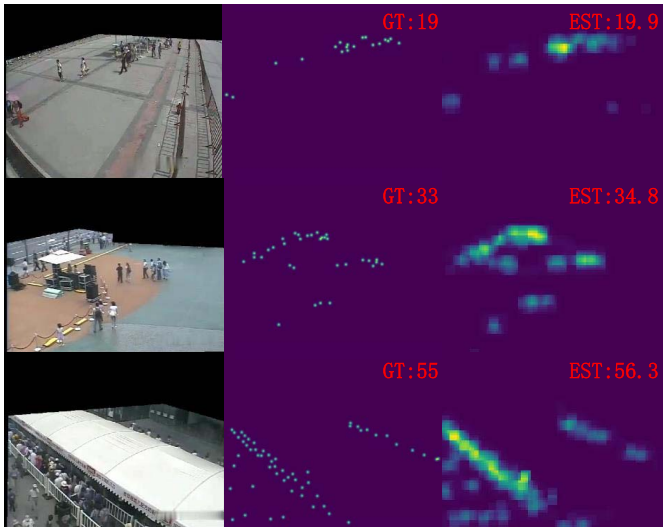


Fig. 11. Qualitative results on the WorldExpo'10 dataset [2]. Row 1 shows the testing images, row 2 the corresponding ground truths, and row 3 our estimated results.

are shown in Fig. 11. The results demonstrate that our method achieves state-of-the-art performance on scene 2, 3, 5, and is on par with other state-of-the-art methods on scene 1, 4 and the overall metric. It is worth noting that our method exhibits big errors (16.8) on Scene4. In Scene4, there are many dense regions with tiny people whose sizes are far smaller than a block (32×32). One example is shown in Fig. 12. These dense patches including tiny people will be downsampled to

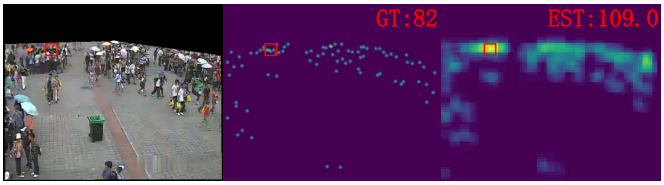


Fig. 12. A failure case on the WorldExpo'10 dataset [2] Scene4. The red rectangle represents a 32×32 block.

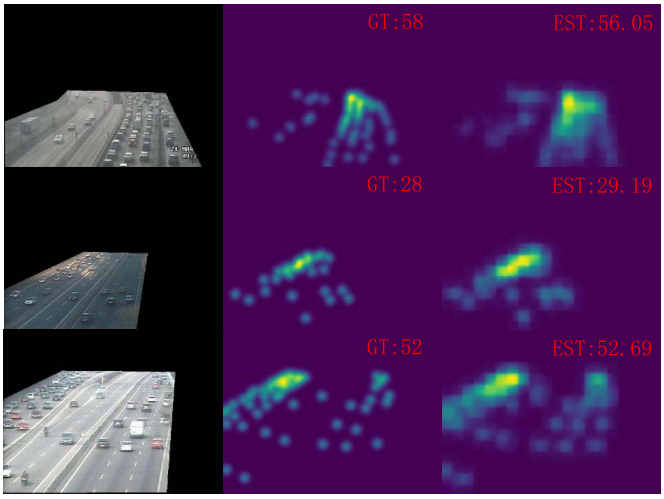


Fig. 13. Qualitative results on the TRANCOS Dataset [20]. Row 1 shows the testing images, row 2 the corresponding ground truths, and row 3 our estimated results.

TABLE XVI
ESTIMATION ERRORS ON THE TRANCOS DATASET

Method	GAME0	GAME1	GAME2	GAME3
Fiaschi et al. [47]	17.77	20.14	23.65	25.99
Lempitsky et al. [7]	13.76	16.72	20.72	24.36
Hydra-3s [28]	10.99	13.75	16.69	19.32
FCN-HA [48]	4.21	——-	——-	———
CSRNET [38]	3.56	5.49	8.57	15.04
LC-PSPNET [49]	3.57	**4.98**	**7.42**	**11.67**
LC-ResFCN [49]	3.32	5.20	7.92	12.57
Ours	**3.15**	5.45	8.34	15.02

a single point after feature extraction (the downsampling rate is 32), and thus lose reliable feature cues for dense regions. This can be viewed as a limitation of our method.

5) TRANCOS Dataset: The TRANCOS dataset [20] is a dataset showing vehicles on roads. It includes 1, 244 images, including 403 training images, 420 validation images, and 421 testing images. A total of 46, 796 dotting labels are included. Each image has a fixed 640×480 resolution, and a region of interest (ROI) that distinguishes the road from the background is provided. We leverage the grid average mean absolute error (GAME) metric to evaluate the performances, which is defined as follows:

$$GAME(L) = \frac{1}{N}\sum_{n=1}^{N}\left(\sum_{l=1}^{4^L}\left|D^l_{I_n} - D^l_{I^{gt}_n}\right|\right), \tag{14}$$

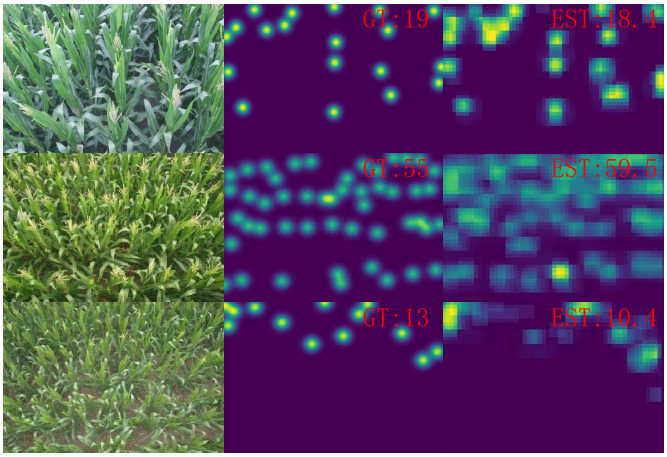


Fig. 14. Qualitative results on the MTC Dataset [12]. Row 1 shows the testing images, row 2 the corresponding ground truths, and row 3 our estimated results.

where N is the number of testing images, $D^l_{I_n}$ is the count for region l in image n, and $D^l_{I^{gt}_n}$ is the corresponding ground-truth. The image regions are obtained by dividing the image into 4^L parts without overlap. A smaller GAME(L) value means the model more accurately describes that local region. Specifically, GAME(0) equals the MAE.

On this dataset, we compared our method with 6 baselines. Reference [48] incorporates a fully convolutional network (FCN) with a long short-term memory network (LSTM) to count the vehicles in the videos. [28] combines FC layers processed from a pyramid of input patches. [49] proposes a detection-based method that focuses only on localizing object instances in the scene. Some result samples are shown in Fig. 13, and the results are listed in Table XVI. As Table XVI shows, our method outperforms the other state-of-the-art methods on GAME0 and achieves parity with the other methods on GAME1 to GAME3.

6) MTC Dataset: MTC Dataset [12] includes 361 images for maize-tassel-counting that were collected from 16 independent time-series image sequences. Of these, 186 images from 8 sequences form the training and validation sets, and 175 images from another 8 sequences form the testing set. This dataset provides dot annotation for each maize tassel. The original image resolution is larger than 3000×2000; therefore, we resized all the images in the training and testing sets $1/8$ of their original size. We compare our method against 6 other state-of-the-art methods. JointSeg [50] is the state-of-the-art tassel segmentation method. mTASSEL [52] uses an object detection approach to count tassels. GlobalReg [51] regresses the global count of tassel images. The results shown in Table XVII illustrate that our method is also effective at counting maize-tassels. Some result samples are shown in Fig. 14.

7) Sonar Fish Dataset: First, we briefly introduce the sonar data. Limited by equipment costs and unusual working conditions such as riverbeds or seabeds, sonar image datasets are scarce and rarely made publicly available. We collected some online sonar video sequences[1] and split them into

[1]http://www.soundmetrics.com/

TABLE XVII
ESTIMATION ERRORS ON THE MTC DATASET

Method	MAE	MSE
JointSeg [50]	24.2	31.6
Hydra [28]	21.0	25.5
GlobalReg [51]	19.7	23.3
mTASSEL [52]	19.6	26.1
DensityReg [7]	11.9	14.8
TasselNet [12]	6.6	9.6
Ours	**5.4**	**9.6**

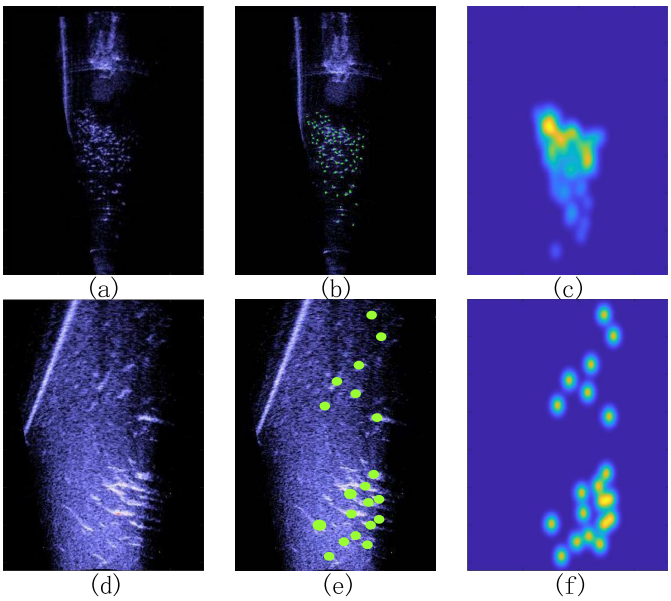


Fig. 15. Sonar images, manually labeled dotted annotations and density map ground-truths: (a) and (d) are the sonar images, where (a) is a dense fish image and (d) is a noisy fish image; (b) and (e) are the annotated labeled images; and (c) and (f) are the ground-truth density maps.

individual images. The fish are annotated based on interframe movement using individual dotted annotations. A total of 537 images from 30 sequences are included in our dataset. The total number of dotting labels is 18,032 among 537 images. A total of 417 images from 24 sequences are used as the training and validation sets, while the remaining 120 images from 6 sequences are used as the test set. This dataset is quite challenging. As shown in Fig. 15, sonar images contain high noise levels and have unstable visual characteristics. In addition, the images are typical: they have low resolution (approximately 360×360), and the image RoIs are considerably smaller than the full images (occupying approximately only one-third of the full image). Thus, the dataset possesses large inhomogeneous distributions, especially in the cropped patches.

The results shown in Table XVIII indicate that our method outperforms the other competitors on the fish-counting task. The performance of [7] may be limited by the feature representation because the features are not tuned in a data-driven manner. The methods of [1] and [12] neglect the issue of

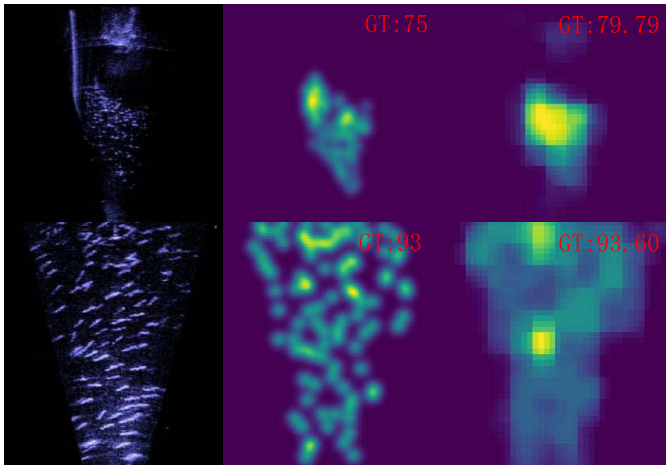


Fig. 16. Qualitative results on the Sonar Fish Dataset. Row 1 shows the testing images, row 2 the corresponding ground truths, and row 3 our estimated results.

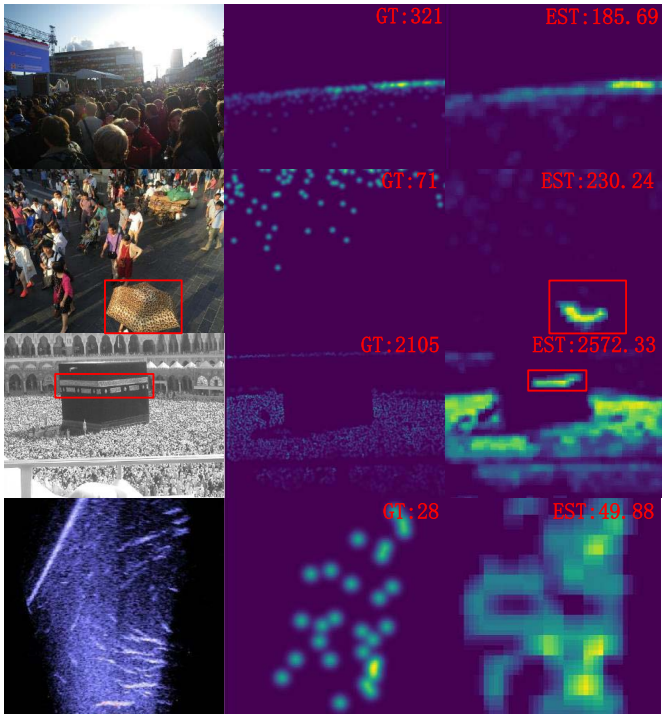


Fig. 17. Failure case samples. The samples in the first and second rows are from ShanghaiTech Part_A and Part_B, the sample in the third row is from UCF_CC_50, and the sample in the last row is from the Sonar Fish dataset.

sample imbalance, which leads to poor performances. Reference [13] attempts to address the sample imbalance in fish counting via a loss function by local regression; thus, it suffers from the disadvantages of regression. Some of the estimated result samples are shown in Fig. 16.

C. Failure Case Analyses

Although our method obtains outstanding performance on the testing benchmarks, it still has some defects. We show some failure cases in this section. First, as shown in the first column in Fig. 17, the massive illumination variation causes an

Fig. 18. Failure cases in ShanghaiTech Part_A (row 1) and Part_B (row 2) for CSRNet [38].

TABLE XVIII
ESTIMATION ERRORS ON THE SONAR FISH DATASET

Method	MAE	MSE
DensityReg [7]	28.17	42.12
MCNN [1]	16.57	24.76
TasselNet [12]	12.89	18.37
Liu et al. [13]	10.98	16.48
Ours	**10.37**	**15.23**

unreliable appearance, leading to poor performance. Second, in crowd counting, the heads present as dots. Our method recognizes other spots as crowds by mistake, as shown in the red frames in Fig. 17, rows 2 and 3. Finally, fish in sonar images are subject to serious noise because of the imaging principle. Our method is less able to distinguish noise from fish, as shown in Fig. 17, row 4.

In addition, we illustrate some failure cases in ShanghaiTech Part_A and Part_B for a state-of-the-art method—CSRNet [38]. As shown in Fig. 18, CSRNet is not only sensitive to illumination variations but also produces fake responses in non-crowd areas.

V. CONCLUSION

In this paper, we introduce the method of object counting by blockwise classification. We analyze the impact of sample imbalance, Gaussian kernel and quantization error in detail and verify the effectiveness of each module in our method. The extensive experiments show that our method outperforms, or at minimum, achieves parity with the state-of-the-art approaches on five object-counting benchmarks.

Our method is not only a tidy and strong baseline for object counting but also has sufficient room for improvement. In future work, building upon the classification framework, we plan to design a new method to achieve better performance that can adapt to scale variations.

REFERENCES

[1] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 589–597.

[2] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-scene crowd counting via deep convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 833–841.

[3] C. Mauricio, F. Schneider, R. Takahira, L. Santos, and H. Gamba, "Image-based red blood cell counter for multiple species of wild and domestic animals," *Arquivo Brasileiro Med. Veterinária Zootecnia*, vol. 69, no. 1, pp. 75–84, 2017.

[4] Y. van Heezik and P. J. Seddon, "Counting birds in urban areas: A review of methods for the estimation of abundance," in *Ecology and Conservation of Birds in Urban Environments*. Cham, Switzerland: Springer, 2017.

[5] B. Wu and R. Nevatia, "Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors," *Int. J. Comput. Vis.*, vol. 75, no. 2, pp. 247–266, Nov. 2007.

[6] M. Wang and X. Wang, "Automatic adaptation of a generic pedestrian detector to a specific traffic scene," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 3401–3408.

[7] V. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2010, pp. 1324–1332.

[8] Z. Shen, Y. Xu, B. Ni, M. Wang, J. Hu, and X. Yang, "Crowd counting via adversarial cross-scale consistency pursuit," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 5245–5254.

[9] L. Zhang, M. Shi, and Q. Chen, "Crowd counting via scale-adaptive convolutional neural network," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1113–1121.

[10] D. Kang and A. B. Chan, "Crowd counting by adaptively fusing predictions from an image pyramid," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*. Newcastle, U.K.: Northumbria Univ., Sep. 2018, p. 89.

[11] J. P. Cohen, G. Boucher, C. A. Glastonbury, H. Z. Lo, and Y. Bengio, "Count-ception: Counting by fully convolutional redundant counting," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2017, pp. 18–26.

[12] H. Lu, Z. Cao, Y. Xiao, B. Zhuang, and C. Shen, "TasselNet: Counting maize tassels in the wild via local counts regression network," *Plant Methods*, vol. 13, no. 1, 2017, Art. no. 79.

[13] L. Liu, H. Lu, Z. Cao, and Y. Xiao, "Counting fish in sonar images," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 3189–3193.

[14] V. A. Sindagi and V. M. Patel, "Generating high-quality crowd density maps using contextual pyramid CNNs," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1879–1888.

[15] Q. Dong, S. Gong, and X. Zhu, "Class rectification hard mining for imbalanced deep learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1851–1860.

[16] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.

[17] W. Shen, X. Wang, Y. Wang, X. Bai, and Z. Zhang, "DeepContour: A deep convolutional feature learned by positive-sharing loss for contour detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3982–3991.

[18] H. Idrees *et al.*, "Composition loss for counting, density map estimation and localization in dense crowds," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 532–546.

[19] H. Idrees, I. Saleemi, C. Seibert, and M. Shah, "Multi-source multi-scale counting in extremely dense crowd images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 2547–2554.

[20] R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. López-Sastre, S. Maldonado-Bascón, and D. Oñoro-Rubio, "Extremely overlapping vehicle counting," in *Proc. Iberian Conf. Pattern Recognit. Image Anal.*, 2015, pp. 423–431.

[21] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.

[22] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–7.

[23] K. Chen, C. C. Loy, S. Gong, and T. Xiang, "Feature mining for localised crowd counting," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2012, p. 3.

[24] L. Lebanoff and H. Idrees, "Counting in dense crowds using deep learning," CRCV, Tech. Rep., 2015.

[25] X. Haipeng, L. Hao, L. Chengxin, L. Liang, C. Zhiguo, and S. Chunhua, "From open set to closed set: Counting objects by spatial divide-and-conquer," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019.

[26] D. B. Sam, S. Surya, and R. V. Babu, "Switching convolutional neural network for crowd counting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, vol. 1, no. 3, pp. 5744–5752.

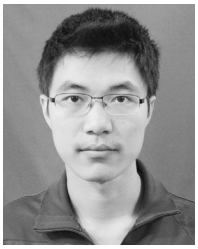
[27] D. Deb and J. Ventura, "An aggregated multicolumn dilated convolution network for perspective-free counting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 195–204.
[28] D. Oñoro-Rubio and R. J. López-Sastre, "Towards perspective-free object counting with deep learning," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 615–629.
[29] Z. Wang, Z. Xiao, K. Xie, Q. Qiu, X. Zhen, and X. Cao, "In defense of single-column networks for crowd counting," 2018, *arXiv:1808.06133*. [Online]. Available: https://arxiv.org/abs/1808.06133
[30] L. Liu, H. Wang, G. Li, W. Ouyang, and L. Lin, "Crowd counting using deep recurrent spatial-aware network," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 849–855.
[31] V. Ranjan, H. Le, and M. Hoai, "Iterative crowd counting," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 270–285.
[32] H. Li *et al.*, "Structured inhomogeneous density map learning for crowd counting," 2018, *arXiv:1801.06642*. [Online]. Available: https://arxiv.org/abs/1801.06642
[33] 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. IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug./Sep. 2017, pp. 1–6.
[34] J. Liu, C. Gao, D. Meng, and A. G. Hauptmann, "DecideNet: Counting varying density crowds through attention guided detection and density estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 5197–5206.
[35] Y. Cao, Z. Wu, and C. Shen, "Estimating depth from monocular images as classification using deep fully convolutional residual networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 11, pp. 3174–3182, Nov. 2017.
[36] R. Li, K. Xian, C. Shen, Z. Cao, H. Lu, and L. Hang, "Deep attention-based classification network for robust depth prediction," in *Proc. Asian Conf. Comput. Vis. (ACCV)*, 2018, pp. 663–678.
[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Sep. 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556
[38] Y. Li, X. Zhang, and D. Chen, "CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 1091–1100.
[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.
[40] B. Li, Y. Dai, and M. He, "Monocular depth estimation with hierarchical fusion of dilated cnns and soft-weighted-sum inference," *Pattern Recognit.*, vol. 83, pp. 328–339, Nov. 2018.
[41] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 4510–4520.
[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[43] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[44] X. Liu, J. van de Weijer, and A. D. Bagdanov, "Leveraging unlabeled data for crowd counting by learning to rank," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 7661–7669.
[45] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
[46] Z. Shi *et al.*, "Crowd counting with deep negative correlation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 5382–5390.
[47] L. Fiaschi, R. Nair, U. Koethe, and F. A. Hamprecht, "Learning to count with regression forest and structured labels," in *Proc. Int. Conf. Pattern Recognit. (ICPR)*, 2012, pp. 2685–2688.
[48] S. Zhang, G. Wu, J. P. Costeira, and J. M. F. Moura, "FCN-rLSTM: Deep spatio-temporal neural networks for vehicle counting in city cameras," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3667–3676.
[49] I. H. Laradji, N. Rostamzadeh, P. O. Pinheiro, D. Vazquez, and M. Schmidt, "Where are the blobs: Counting by localization with point supervision," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 547–562.
[50] H. Lu, Z. Cao, Y. Xiao, Y. Li, and Y. Zhu, "Region-based colour modelling for joint crop and maize tassel segmentation," *Biosyst. Eng.*, vol. 147, pp. 139–150, Jul. 2016.
[51] K. Tota and H. Idrees, "Counting in dense crowds using deep features," in *Proc. CRCV*, 2015, pp. 1–4.
[52] H. Lu, Z. Cao, Y. Xiao, Z. Fang, Y. Zhu, and K. Xian, "Fine-grained maize tassel trait characterization with multi-view representations," *Comput. Electron. Agricult.*, vol. 118, pp. 143–158, 2015.

Liang Liu received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2016, where he is currently pursuing the Ph.D. degree with the School of Artificial Intelligence and Automation.

His research interests include computer vision and machine learning with particular emphasis on object counting and various computer vision applications in agriculture.

Hao Lu received the Ph.D. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2018.

He is currently a Post-Doctoral Fellow with the School of Computer Science, The University of Adelaide. His research interests include computer vision, image processing, and machine learning. He has worked on topics including visual domain adaptation, fine-grained visual categorization, and miscellaneous computer vision applications in agriculture. His current interests are object counting and dense labeling problems, such as low-level image matting, image denoising and high-level semantic segmentation, and depth estimation.



Haipeng Xiong received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2018, where he is currently pursuing the M.S. degree with the School of Artificial Intelligence and Automation.

His research interests are object counting and its applications in agriculture.

Ke Xian received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2014, where he is currently pursuing the Ph.D. degree with the School of Artificial Intelligence and Automation.

His current research interests primarily centers on algorithms issues in computer vision and deep learning, including depth estimation from single images, semantic image segmentation, and 2D-to-3D conversion.

Zhiguo Cao received the B.S. and M.S. degrees in communication and information system from the University of Electronic Science and Technology of China, Chengdu, China, and the Ph.D. degree in pattern recognition and intelligent system from the Huazhong University of Science and Technology, Wuhan, China.

He is currently a Professor with the School of Artificial Intelligence and Automation, Huazhong University of Science and Technology. He has authored dozens of articles at international journals and conferences, which have been applied to automatic observation system for object recognition in video surveillance system, for crop growth in agriculture and for weather phenomenon in meteorology based on computer vision. His research interests include spread across image understanding and analysis, depth information extraction, and object detection.

Dr. Cao's projects have received provincial or ministerial level awards of Science and Technology Progress in China.

Chunhua Shen studied at Nanjing University and at Australian National University. He received the Ph.D. degree from The University of Adelaide, Adelaide, Australia.

He is currently a Professor with the School of Computer Science, The University of Adelaide. His research interests are in the intersection of computer vision and statistical machine learning. In 2012, he was awarded the Australian Research Council Future Fellowship.