Cells Counting with Convolutional Neural Network

Run-xu Tan[1], Jun Zhang[1(✉)], Peng Chen[2], Bing Wang[3], and Yi Xia[1]

[1] College of Electrical Engineering and Automation, Anhui University, Hefei 230601, Anhui, China
wwwzhangjun@163.com
[2] Institute of Health Sciences, Anhui University, Hefei 230601, Anhui, China
[3] School of Electrical and Information Engineering, Anhui University of Technology, Ma Anshan 243032, China

Abstract. In this paper, we focus on the problem of cells objects counting. We propose a novel deep learning framework for small object counting named Unite CNN (U-CNN). The U-CNN is used as a regression model to learn the characteristics of input patches. The result of our model output is the density map. Density map can get the exact count of cells, and we can see the location of cell distribution. The regression network predicts a count of the objects that exit inside this frame. Unite CNN learns a multiscale non-linear regression model which uses a pyramid of image patches extracted at multiple scales to perform the final density prediction. We use three different cell counting benchmarks (MAE, MSE, GAME). Our method is tested on the cell pictures under microscope and shown to outperform the state of the art methods.

Keywords: Deep learning · Cell counting · Unite CNN

1 Introduction

Cell counting has many uses in medicine. Some routine tests such as the Clinical test of hospital, inflammation of the body, need to count the number of white blood cells in the blood, and another cell related research will be used. The classical approach to counting involves fine-tuning edge detectors to segment objects from the background [1] and counting each one. A large challenge here is dealing with overlapping objects which require methods such as the watershed transformation [2]. These approaches have many hyperparameters specifically for each task and are complicated to build. We designed a new network model, and trained on the Linux system to get the model for counting.

Using the principle and method of deep learning, we first designed a separate network structure, which has five convolutional layers and four average pooling layers. The cell data into two parts, one is the training set and the other is the test set. The training set includes 40 cell pictures, and the test set is 10 cell pictures. Although the regression model of our individual network is fast in counting, the accuracy of counting needs to be improved. In order to solve this problem, we proposed a Unite CNN model by amplifying training data set and adjusting the network parameters to optimize the

D.-S. Huang et al. (Eds.): ICIC 2018, LNAI 10956, pp. 102–111, 2018.
https://doi.org/10.1007/978-3-319-95957-3_12

network structure; through the optimization of the network, we get the best network model, and then use the method of cross validation [3] to verify the accuracy and stability of the system.

1.1 Related Work

Deep learning is a new method of image recognition in recent ten years. It originates from artificial neural network, it is a deep network structure. Deep learning has great advantages in the field of image recognition. For example, Hinton and his students at the ImageNet competition in 2012 [4], They used a convolutional neural network in deep learning to do a picture classification experiment. In this experiment, they need to divide one million pictures into one thousand classes. The error rate of the experimental classification is 15%, which is nearly 11 percentage points higher than that of second. It fully proved the superiority of deep learning. Compared with the traditional identification methods, the advantages of deep learning are that it can obtain the sample features from large amounts of data automatically, reduce the manual design steps. The final output model is obtained by continuous iteration and extraction of the characteristics of the sample.

Various methods have been proposed to traditional image counting. These method can be divided into the following categories: Detection-based method [5], Regression-based method [6] and Density estimation-based method. Here we focus on reviewing papers based on regression model counting, because our method is also part of the group. These methods define the mapping from the coordinate map of the input image to the object count. *Lempitsky et al.* introduces a counting method [7], which learns from the linear mapping of the image local feature to the density map of the object. By successful learning, we can provide object counting by simply integrating multiple regions in the estimated target density map.

We design network models by deep learning, which considers the problem of cell counting as the task of object density estimation. This method is quite different from traditional methods. We also see some good models and algorithms in the crowd counting, for example *Zhang et al*. [8] proposed a method to predict the density map with the structure of CNN, and they used two different loss function for training a switchable learning process. They extracted features using different size of filters and combined them as final count estimates. We used a simpler way to train our model, and we got better results.

2 Proposed Method

2.1 Cells Counting Model

The data we used was a red blood cell image observed under a microscope. In the experiment, we used 50 pictures of the original cells to train the network, and the number of cells in each picture is about 200. At the same time before training, we need to annotate the cells in the raw images. Annotated images provide the coordinates of

the cells in the network training process, and they can be used as ground truth to ensure the accuracy of network learning.

There are five parts of our network structure. The first part and the second part respectively have two convolutional layers, the size of the convolution filter is 3 * 3, the number of convolution kernels in the first part is 64, and the number of convolution kernels in the second part is 128. The third part to the fifth part consists of three convolutional layers, the size of the convolution filter is also 3 * 3, the number of convolution filters in the third part is 256, and the number of convolution filters in the fourth and fifth parts is 512. For the four parts, A pooling layer is added behind each part. In order to ensure that the count will not cause loss because the max-pooling layer, we add the average pooling layers to the network structure. All the previous layers are followed by rectified linear units. Our Cells Counting network model is displayed in Fig. 1.

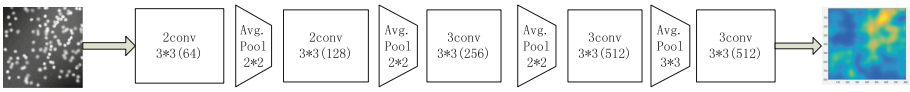

Fig. 1. Our novel cells counting model. The images of the input cells output the corresponding density map through the network, and the cell count is realized by the density map.

We propose an improved method that other people also have done, *Zhang et al.* introduced the new Crowd CNN count structure [13]. We conducted a comparison of the two methods. First of all, in our model, a convolutional layer and the nonlinear activation structure of alternating layers, can extract the deep features better than the single layer structure convolutional. Second, assuming that all data has C channels, then the individual 7 * 7 convolutional layer will contain 7 * 7 * c = 49c2 parameters, while the three 3 * 3 convolutional layers have the combination of 3 * (3 * 3 * c) = 27c2 parameters. Intuitively, it is better to choose a convolutional layer with a small filter instead of a convolutional layer with a large filter. The former can express more powerful features of input data, and less parameters are used. The only downside is that in the reverse propagation, the middle convolutional layer may cause more memory to be occupied. So the size of the convolution filters we used is 3 * 3. We do not need any extra regressor, our cells counting model is learned in an end-to-end manner to predict the object density maps directly. Finally, our experimental results will be shown in the Sect. 4.

2.2 Unite CNN

In the use of a single cell counting model, the input features need to be geometrically corrected, using annotated cell coordinates as ground truth. Through the iterative learning characteristics of the cell images, and loss rate is reducing constantly. The results of this part are shown in the fourth part. We used two methods to evaluate the results of the experiment, it called absolute error (MAE) and mean squared error (MSE), which are defined as follows:

$$MAE = \frac{1}{N}\sum_{1}^{N}|z_i - \hat{z}_i|, MSE = \sqrt{\frac{1}{N}\sum_{1}^{N}(z_i - \hat{z}_i)^2} \quad (1)$$

Where N is the number of test images, z_i is the actual number of cells in the ith image, and $\hat{z}_i$ is the estimated number of cells in the ith image. Roughly speaking, MAE indicates the accuracy of the estimates, and MSE indicates the robustness of the estimates.

Technically, the perspective distortion of image display leads to the features extracted from the same object, but there is great value difference in the depth of the different scenes. Therefore, the model with a single regression function will produce erroneous results. In order to solve this problem, we propose Unite CNN. We set the stride of the fourth avg-pooling layer to 1. The resolution of the output picture of the network is 1/8 of the resolution of the input picture. We use the hole technique to deal with the mismatch of the receiving field caused by the removal of the stride in the fourth avg-pooling layers. The convolution filter with holes can have any large receiving domain, regardless of the size of its kernel. Using holes, we double the receptive field of convolutional layers after the fourth avg-pool layer, thereby enabling them to operate with their originally trained receptive field. As illustrated in Fig. 2. Our model uses two ways, two head and three head (UCNN-h2, UCNN-h3). Each head uses Cells Counting model, then the outputs of the different heads are concatenated and passed to the three full convolutional layers, with 512 neurons each one, which are followed by a ReLU and a dropout layer. Our architecture uses a fully connected layer FC8 as the end, which has 324 neurons. The result of the final output is the density map of the cells. To train UCNN model we use the loss function defined in Eq. (2):

$$l(\Theta) = \frac{1}{2N}\sum_{n=1}^{N}\left\|R(P_n|\Theta) - D_{gt}^{(P_n)}\right\|_2^2 \quad (2)$$

Where N represents the number of training images being divided, and $D_{gt}^{(P_n)}$ represents the ground-truth density for associated training patch P_n. Recall that Θ encodes the network parameters.

$$\mathrm{GAME(L)} = \frac{1}{\mathrm{N}}\sum\nolimits_{\mathrm{n}=1}^{\mathrm{N}}\left(\sum\nolimits_{\mathrm{l}=1}^{4^{\mathrm{l}}}\left|\mathrm{D}_{\mathrm{I}_{\mathrm{n}}}^{\mathrm{l}} - \mathrm{D}_{\mathrm{I}_{\mathrm{n}}^{\mathrm{gt}}}^{\mathrm{l}}\right|\right) \quad (3)$$

Where N is the total number of images, $D_{I_n}^{l}$ corresponds to the estimated object density map count for the image n and region l, and $D_{I_n^{gt}}^{l}$ is the corresponding ground truth density map. For a specific level L, the GAME(L) subdivides the image using a grid of 4^{L} non-overlapping regions, and the error is computed as the sum of the mean absolute errors in each of these subregions. This metric provides a spatial measurement of the error. Note that a GAME(0) is equivalent to the mean absolute error (MAE).

For visual significance estimation, our network design uses Hydra CNN [9] and Deep network [10] for reference. They put forward a different network structure, using

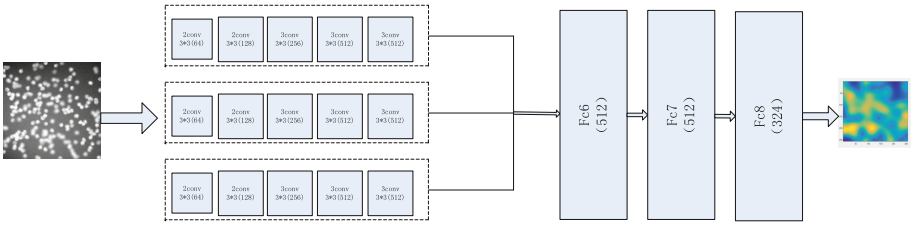

Fig. 2. Unite CNN, our network is designed to enhance the robustness of the scale change by training the patch of the multiscale image in pyramid.

a multiple input strategy. It combines the features of different views of the whole input image to return a visual saliency map. In our UCNN model, the output from layer is upsampled to the size of the input image using bilinear interpolation to obtain the final cells density map. In order to achieve the final count, we can get the total count in the image by predicting the density map. Finally, our experimental results will be shown in the Sect. 4 (Fig. 3).

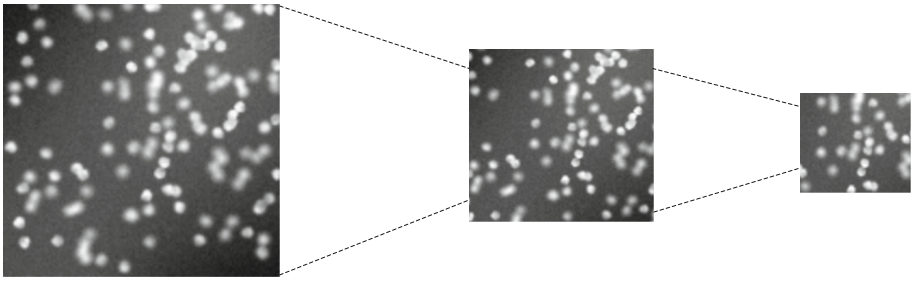

Fig. 3. Our experimental data will be processed pyramid before input, forming different sizes of patch, so that we can expand data volume and increase the stability of the experimental results.

3 Experiments

We evaluate our UCNN model on red cells dataset under microscope. We first need to process each cell and locate the location of each cell in the original picture to get dotted map. It has achieved competitive and superior performance in our cells dataset. In the end, we also demonstrate the generalizability of such a simple model in the transfer learning setting. Implementation of the proposed network and its training are based on the Caffe framework [15] developed by source code.

In a manner similar to recent works [11], we evaluate the performance of our approach using 5-fold cross validation. We randomly divide the dataset into five splits with each split containing 10 images for test, and 40 images for training. In each fold of the cross validation, we consider five splits (40 images) for training the network and the remaining split (10 images) for validating its performance. For training our models, we scale the images in order to make the largest size equal to 800 pixels. To report the

results the MAE and the MSE are used. We randomly extract 1200 image patches of 150×150 pixels with their corresponding ground truth. We also augment the training data by flipping each sample. To do the test, we densely scan the image with a stride of 10 pixels. We generate the ground truth object density maps with the code provided in [12], which places a Gaussian Kernel(with a covariance matrix of $\sum = 15 \cdot 1_{2x2}$) in the center of each annotated object (Fig. 4).

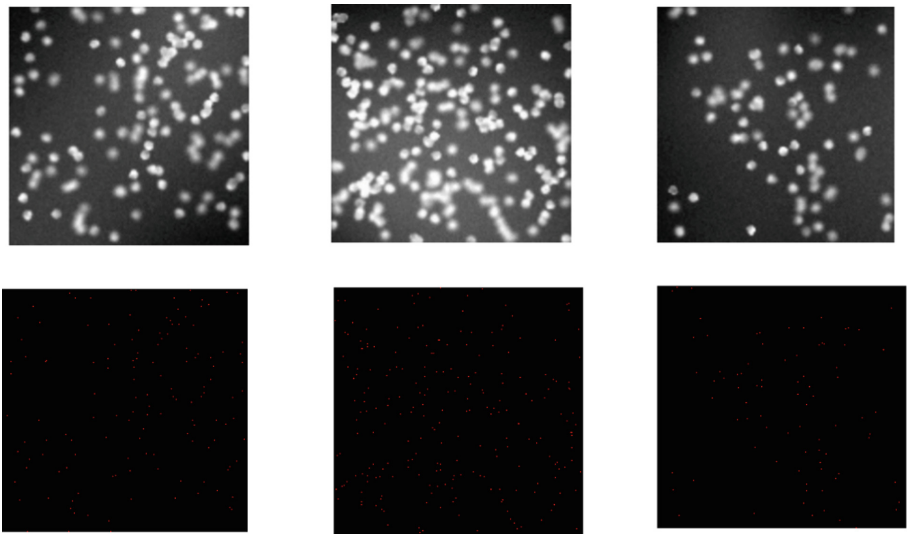

Fig. 4. The picture shows the original map and the dotted map, and the second columns are annotated as the ground truth in the training.

3.1 Cells Counting Results

We show the true results of ten pictures on the test and experimental results in Tables 1, 2 and 3, and the time for our three network framework of the experiment. We can see obviously when the network framework becomes complicated, in the same 50000 iterations of the case, it will increase the training time.

The Tables 1, 2 and 3 picture of the true count and prediction count were compared. Figure 5. shows the three methods we proposed the predictive value of fitting with the actual value. We got the first and third network (Cells Counting model and UCNN-h3) framework of the error was very small and the predicted values and the real value was close, but the network framework second (UCNN-h2) appeared a great error, the experiment effect is not good. This estimation error could possibly be a consequence of the insufficient number of training images with such large cells in the dataset.

3.2 Experiments Analysis

In this section, we have an overall analysis of all the data of the experiment. The criteria used include MAE, MSE, and GAME0-3.

Table 1.

Cells counting model			
Image0	True = 122.00	Pred = 128.15	Time = 16.44 s
Image1	True = 206.00	Pred = 207.09	Time = 16.39 s
Image2	True = 300.00	Pred = 292.70	Time = 16.37 s
Image3	True = 196.00	Pred = 197.26	Time = 16.38 s
Image4	True = 204.00	Pred = 209.78	Time = 16.39 s
Image5	True = 168.00	Pred = 171.28	Time = 16.41 s
Image6	True = 168.00	Pred = 170.23	Time = 16.60 s
Image7	True = 129.00	Pred = 140.22	Time = 16.37 s
Image8	True = 197.00	Pred = 194.42	Time = 16.39 s
Image9	True = 223.00	Pred = 220.18	Time = 16.37 s

Table 2.

UCNN-h2			
Image0	True = 122.00	Pred = 163.32	Time = 32.75 s
Image1	True = 206.00	Pred = 208.61	Time = 32.77 s
Image2	True = 300.00	Pred = 252.62	Time = 32.77 s
Image3	True = 196.00	Pred = 201.66	Time = 32.84 s
Image4	True = 204.00	Pred = 210.52	Time = 32.77 s
Image5	True = 168.00	Pred = 188.46	Time = 32.87 s
Image6	True = 168.00	Pred = 187.90	Time = 32.79 s
Image7	True = 129.00	Pred = 171.58	Time = 32.75 s
Image8	True = 197.00	Pred = 199.89	Time = 32.76 s
Image9	True = 223.00	Pred = 214.87	Time = 32.82 s

Table 3.

UCNN-h3			
Image0	True = 122.00	Pred = 125.64	Time = 46.83 s
Image1	True = 206.00	Pred = 206.96	Time = 46.86 s
Image2	True = 300.00	Pred = 292.30	Time = 46.72 s
Image3	True = 196.00	Pred = 198.10	Time = 46.95 s
Image4	True = 204.00	Pred = 207.45	Time = 46.75 s
Image5	True = 168.00	Pred = 171.44	Time = 47.09 s
Image6	True = 168.00	Pred = 168.64	Time = 46.75 s
Image7	True = 129.00	Pred = 140.07	Time = 47.76 s
Image8	True = 197.00	Pred = 194.69	Time = 46.75 s
Image9	True = 223.00	Pred = 220.70	Time = 46.77 s

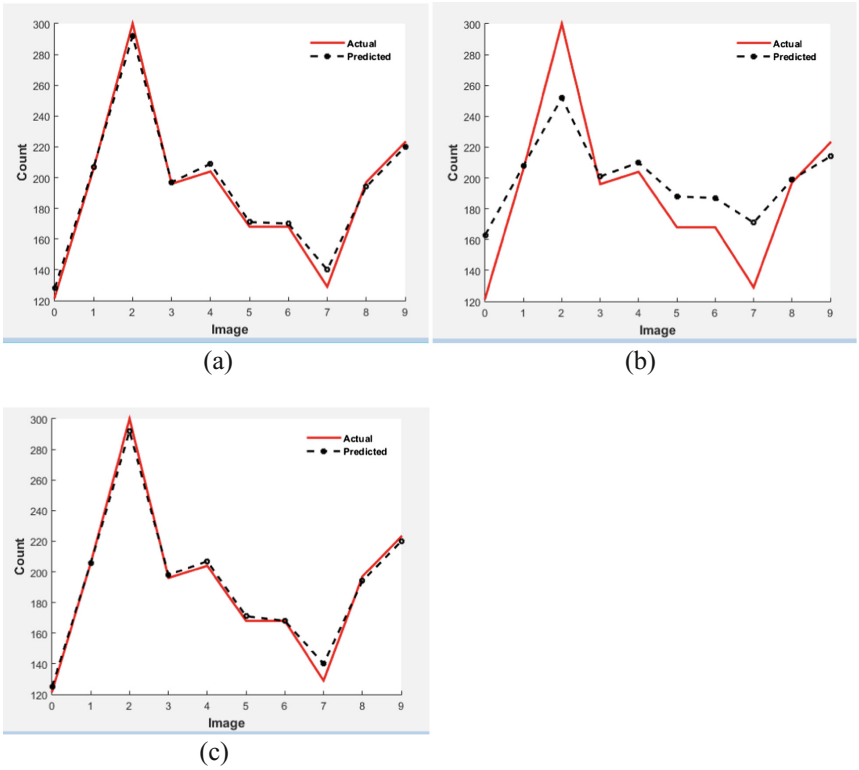

Fig. 5. Actual count vs. prediction count for cells dataset. (a) (b) (c) show the results of cells counting model, UCNN-h2, and UCNN-h3 respectively.

Table 4 shows the experimental results of the three methods we proposed at the same evaluation standard. Under this data, the best result is our UCNN-h3, which effectively reduces the value of MAE. Analyzing the results, We can see that when the angle of view is constant, the cell count error will be larger when the cell density is big. With regard to the cell count and the results provided, our network model has a certain advantage in counting the red cell dataset under the microscope.

Table 4.

Method	MAE	MSE	GAME0	GAME1	GAME2	GAME3
Cell Counting model	7.31	8.74	4.37	24.36	34.31	39.58
UCNN-h2	47.66	54.68	19.78	36.01	45.65	55.24
UCNN-h3	6.72	8.05	3.76	23.57	33.30	38.45

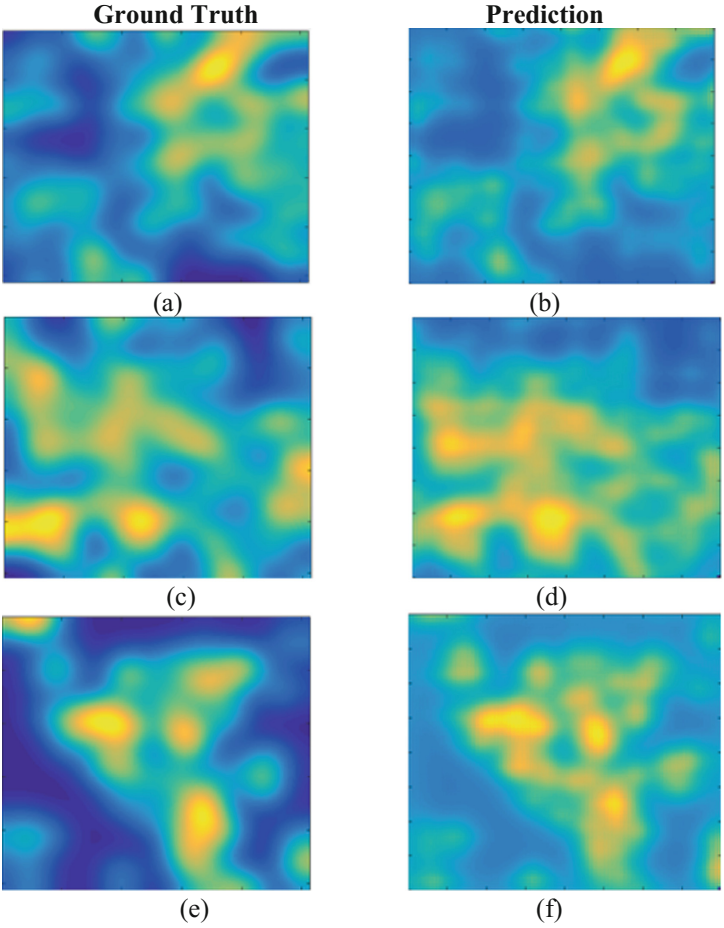

Fig. 6. Qualitative results of the UCNN-h3 in the cells dataset. (a) (c) (e) represent the density of the cells' true marked. (b) (d) (f) represent the predicted density map of the experiment.

Figure 6 shows some quantitative results. These results are obtained by our UCNN-h3 and are the best results of our experiments. The first column is our ground truth density map, and the second column is the predicted density map.

4 Conclusion

In this paper, we propose a network framework based on deep learning that can be used to count high density cell images. We use the Vgg network [14] and make a multi - column combination. Cell Counting model and UCNN-h3 have little error in cell image counting. At the same time, we also show that there are many problems to be solved in the high-density image counting, and the expansion of data volume can also improve

the experimental results effectively. At present, our method has achieved good results in our own dataset.

Acknowledgement. This work is supported by Anhui Provincial Natural Science Foundation (grant number 1608085MF136).

References

1. Sezgin, M., Sankur, B.: Survey over image thresholding techniques and quantitative performance evaluation. J. Electron. Imaging **13**(1), 146–166 (2004)
2. Beucher, S.: Watershed, hierarchical segmentation and waterfall algorithm. In: Serra, J., Soille, P. (eds.) Mathematical Morphology and Its Applications to Image Processing, pp. 69–76. Springer, Dordrecht (1994). https://doi.org/10.1007/978-94-011-1040-2_10
3. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. J. Mach. Learn. Res. **9**, 249–256 (2010)
4. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: International Conference on Neural Information Processing Systems, vol. 60, pp. 1097–1105. Curran Associates Inc. (2012)
5. Lempitsky, V.S., Zisserman, A.: Learning to count objects in images. In: International Conference on Neural Information Processing Systems, vol. 43, pp. 1324–1332. Curran Associates Inc. (2010)
6. Pham, V.Q., Kozakaya, T., Yamaguchi, O., Okada, R.: COUNT forest: co-voting uncertain number of targets using random forest for crowd density estimation. In: IEEE International Conference on Computer Vision, pp. 3253–3261. IEEE (2015)
7. Lempitsky, V.S., Zisserman, A.: Learning to count objects in images. In: International Conference on Neural Information Processing Systems, vol. 43, pp. 1324–1332. Curran Associates Inc. (2010)
8. Zhang, C., Li, H., Wang, X., Yang, X.: Cross-scene crowd counting via deep convolutional neural networks. In: Computer Vision and Pattern Recognition, pp. 833–841. IEEE (2015)
9. Oñoro-Rubio, D., López-Sastre, R.J.: Towards perspective-free object counting with deep learning. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 615–629. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_38
10. Boominathan, L., Kruthiventi, S.S.S., Babu, R.V.: CrowdNet: a deep convolutional network for dense crowd counting, pp. 640–644 (2016)
11. Idrees, H., Saleemi, I., Seibert, C., Shah, M.: Multi-source multi-scale counting in extremely dense crowd images. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 9, pp. 2547–2554. IEEE Computer Society (2013)
12. Guerrero-Gómez-Olmedo, R., Torre-Jiménez, B., López-Sastre, R., Maldonado-Bascón, S., Oñoro-Rubio, D.: Extremely overlapping vehicle counting. In: Paredes, R., Cardoso, J.S., Pardo, X.M. (eds.) IbPRIA 2015. LNCS, vol. 9117, pp. 423–431. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19390-8_48
13. Zhang, Y., Zhou, D., Chen, S., Gao, S., Ma, Y.: Single-image crowd counting via multi-column convolutional neural network. In: Computer Vision and Pattern Recognition, pp. 589–597. IEEE (2016)
14. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Comput. Sci. (2014)
15. Jia, Y., Shelhamer, E., Donahue, J., et al.: Caffe: convolutional architecture for fast feature embedding, pp. 675–678 (2014)