

Crowd Counting Based on MMCNN in Still Images

Tao Wang¹, Guohui Li¹(✉), Jun Lei¹, Shuohao Li¹, and Shukui Xu²

¹ College of Information System and Management,
National University of Defense Technology, Changsha, China
{wangtao1993,guohli,leijun1987,lishuohao}@nudt.edu.cn

² The 28th Research Institute of China Electronic Technology
Group Corporation, Nanjing, China
xskgfk@163.com

Abstract. Accurately estimate the crowd count from a still image with arbitrary perspective and arbitrary crowd density is one of the difficulties of crowd analysis in surveillance videos. Conventional methods are scene-specific and subject to occlusions. In this paper, we propose a Multi-task Multi-column Convolutional Neural Network (MMCNN) architecture for crowd counting and crowd density estimation in still images of surveillance scenes. The MMCNN architecture is an end-to-end system which is robust for images with different perspective and different crowd density. By promoting MCNN with 3×3 filter, the MMCNN could utilize local spatial features from each column. Furthermore, the ground truth density map is generated based on Perspective-Adaptive Gaussian kernels which can better represent the heads of pedestrians. Finally, we use an iterative switching process in our deep crowd model to alternatively optimize the crowd density map estimation task and crowd counting task. We conduct experiments on the WorldExpo'10 dataset and our method achieves better results.

Keywords: Convolutional neural networks · Crowd counting · Crowd density estimation · MMCNN

1 Introduction

Counting crowd pedestrians in surveillance videos attracts a lot of attention in public safety. It is especially significant in major cities where there are public rallies and sports events. In the new year eve of 2015, 35 people died of stampede in Shanghai, China. Unfortunately, there are many more similar disasters take place around the world. Accurately estimating crowds from images or videos is a highly valued problem of computer vision. In practical applications, crowd counting and density estimation are challenging, because of severe occlusions, diverse crowd distributions and scene perspective distortions.

G. Li—Thanks to my tutor for giving me a lot of support and help.

The most intuitive method is detecting and tracking all the people or foreground segmentation, but these methods are indispensable in practical crowd scenes. Many excellent methods [1–4] were constructed based on regression. These methods firstly represent the crowd scenes into feature space, then learn a mapping from the feature space to crowd counts. However, these works used hand-crafted features which is designed by experienced researchers/engineers. And the these features are scene-specific. Cong Zhang *et al.* [5] proposed a framework for crowd counting based on deep convolutional neural networks, their method uses CNN to extract deep features instead of hand-crafted features. A good method of crowd counting can also be extended to other domains, such as counting cells or bacteria from microscopic images, animal crowd estimates in wild scenes, or estimating the number of cars at traffic jams, etc. [5] constructs a new training set (sampled from source domain) which follows the distribution of target domain to adapt the model to the new scenario. But this work is not an end-to-end system, that is, we must crop the image to a same size set in advance and splice them together when counting the crowd. Yingying Zhang *et al.* [6] proposed a Multi-column Convolutional Neural Network to predict crowd scenes' density maps. But their network is very difficult to obtain optima due to its multi-column structure and large-number-pixels regression task.

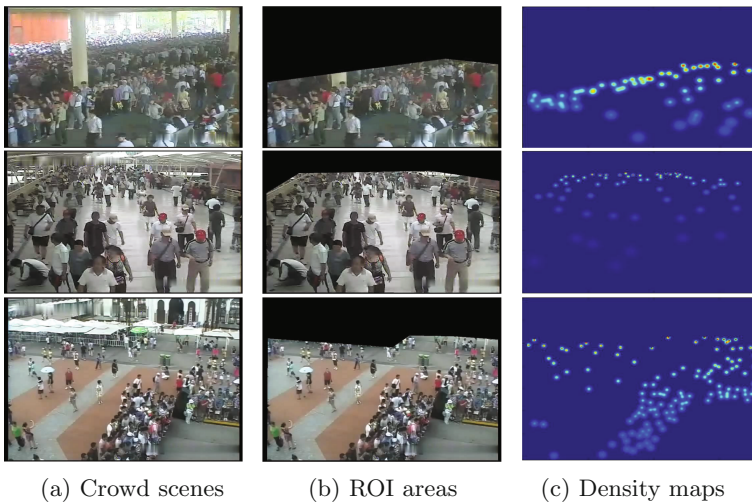


Fig. 1. (a) Crowd scenes there are many occlusions. All the images are selected from the WorldExop'10 dataset [5,37]. (b) ROI areas of crowd scenes because there are much noise. The area out of ROI is set to zero. (c) Density maps generated by using Perspective-adaptive kernels. The size and shape of each kernel is similar to the heads of pedestrians. These density maps are labels of MMCNN when we training crowd density estimation task.

Contribution of This Paper. Inspired by state-of-the-art method [6], we propose a Multi-task Multi-column Convolutional Neural Network for crowd density estimation and counting. We especially put the focus on density estimation and the target is generating a density map close to the reality. By this density map, we can conveniently find the high density areas and warning in advance. Contributions of this paper are summarized as follows:

- (1) We proposed Perspective-Adaptive method to generate head-shaped and head-sized Gaussian Kernels. The Gaussian kernels used in [5] represent the pedestrian contour (head and body) but it does not work well enough when the occlusion is serious. [6] used round Gaussian kernels to describe pedestrians' heads. However, it is not accurate enough when there is perspective distortion.
- (2) Based on MCNN [6], we improve the method of merging the feature maps from different columns with 3×3 filter. In this way, the network could consider local spatial information of each feature map.
- (3) We proposed a Multi-task Multi-column Convolutional Neural Network (MMCNN) based on MCNN [6]. We use an iterative switching process in our deep crowd model to alternatively optimize the density map estimation task and the count estimation task. In this way, the two different but related task can alternatively assist each other to obtain better local optima and make up for the shortcomings of regression framework.

2 Related Work

Many algorithms have proposed in the literature for crowd counting. Earlier methods [7] adopt a detection-style framework to estimate the number of pedestrians. [8–10] have used a similar detection-based framework for pedestrian counting. In detection-base crowd counting methods, people typically assume a crowd is composed of individual entities which can be detected by some given detectors [11–14]. The limitation of such methods is that occlusion among people in a very dense crowd significantly affects the performance of the detector.

In counting crowds in videos, people have proposed to cluster trajectories of tracked visual features. Such as [15, 16]. But these tracking-based methods do not work for estimating crowds from individual still images.

The most extensively used method for crowd counting is feature-based regression, see [17–22]. The main steps of these methods are: (1) extracting the foreground; (2) extracting various features from the foreground, such as area of crowd mask [17, 18, 20, 23], edge count [16–18, 24], for texture features [17, 25]; (3) utilizing a regression function to estimate the crowd count. Linear [23] or piece-wise linear [24] functions are relatively simple models and get decent performance. Other more advanced methods are ridge regression (RR) [17], Gaussian process regression (GPR) [16], and neural network (NN) [25].

There have also been some works focusing on crowd counting from still images. [26] has proposed to leverage multiple sources of information to compute

an estimate of the number of individuals present in an extremely dense crowd. In that work, a dataset of fifty crowd images containing 64K annotated humans (UCF_CC_50) is introduced. [27] has followed the work and estimated counts by fusing information of multiple sources, namely, interest points (SIFT), Fourier analysis, wavelet decomposition, GLCM features, and low confidence head detections. [28] has utilized the features extracted from a pre-trained CNN to train a support vector machine (SVM) that subsequently generates counts for still images.

Many works introduce deep learning into various surveillance applications, such as person re-identification [29], pedestrian detection [1, 30, 31], tracking [32], crowd behavior analysis [33] and crowd segmentation [34]. Their success benefits from discriminative power of deep models. Zhang *et al.* [5] has proposed a CNN based method to count crowd in different scenes. They first pre-train a network for certain scenes. When a test image from a new scene is given, they choose similar training data to fine-tune the pre-trained network based on the perspective information and similarity in density map. Their method demonstrates good performance on most existing datasets. But this type of methods need to crop one image into patches of similar sizes, then estimate the total number of each patch, and merge all the patches into one image. In this way, there are inevitably overlaps between patches. And this method is not an end-to-end framework, it is not suitable for practical application such as real-time public safety monitoring. Yingying Zhang *et al.* [6] recently proposed a Multi-column Convolutional Neural Network. They use three columns of CNNs corresponding to filters with receptive fields of different sizes so that the features learned by each column CNNs could deal with perspective effect and different image resolution. But they did not take perspective distortion of heads into consideration.

3 Method

3.1 Perspective-Adaptive Kernels

Many works followed [5] and defined the density map regression ground truth as a sum of Gaussian kernels centered on the location of objects. This kind of density map is suitable for characterizing the density distribution of circle-like objects such as cells and bacteria. [5] uses human-shaped Gaussian kernels to generate density map. But due to severe occlusions (Fig. 1(a)), heads are the main cues to judge whether there exists a pedestrian in a practical surveillance scene. [6] uses circle-like Gaussian kernels to represent the heads of pedestrians in the image. It works when the scene is full of pedestrians but actually the heads in the image are not circle-like because of perspective distortion. Precisely, the shapes of heads are more similar to ellipses than circles. We proposed a method of generating density maps based on Perspective-Adaptive Kernels:

Firstly, we generate each scene's perspective map M by linear regression [5] or geometry-adaptive method [6]. Most of the time, the datasets have provided the perspective map, such as the WorldExpo'10 dataset. The value of each element in the perspective map $M(p)$ denotes the number of pixels in the image representing

one square meter at that location in the actual scene. The value of M contains the perspective information. For example, if the same person is standing at two point with different distances from the camera, it will look shorter when he is in the place far away from the camera, the head will be smaller, and the $M(p)$ value will be smaller too, vice versa. After we obtain the perspective map and the center positions of pedestrian head p_i in the region of interest (ROI), we create the crowd density map as:

$$D_i(p) = \sum_{P \subseteq P_i} \frac{1}{\|Z\|} (N_h(p; P_h, \Sigma)) \quad (1)$$

where g is the gradient of perspective map, C_1 and C_2 are empirical coefficients, N_h represents Gaussian kernel and $\Sigma = (\sigma_x, \sigma_y)$, P_h represents the position of head. Assume that perspective distortion only exists in y-axis and the heads in the image is distorted only in y-axis. To exactly represent the pedestrian head, we set the variance $\sigma_x = C_1 M(p)$, $\sigma_y = (1 + gC_2)\sigma_x$.

3.2 Normalized Crowd Density Map

Our MMCNN model has two switchable learning objectives, the main task is to estimate the crowd density map of the input image. Because density map represents the distribution of pedestrians and has abundant spatial information. To ensure that the integration of all density values in a density map equals to the total crowd number in the original image, the whole distribution is normalized by Z . The created density maps are showed in Fig. 1(c). To obtain density maps precisely, we conduct a lot of experiments and find the best empirical coefficients, $C_1 = 0.12$, $C_2 = 0.5$. At this time, the size of Gaussian kernel is close to pedestrian head size in the image and will not appear too small. We only consider the pedestrians in ROI and truncate the Gaussian distribution outside ROI.

3.3 Multi-task Multi-column CNN Model

An overview of our crowd MMCNN model is shown in Fig. 2. Our MMCNN model uses the base framework of MCNN [6]. The input image is original still image from surveillance video. In our model, we use the filters of different sizes to perceptive heads of different scales and there is no need to crop the image into a pre-set size like [5]. Motivated by the good performance Fully Convolutional Networks [36], we promote the MCNN [6] model by using 3×3 filters instead of 1×1 filters to merge the *Concated feature map* into one-channel feature map. Suppose that the shape of *Concated feature map* volume is $42 \times 144 \times 180$, if we use 1×1 filters to merge all the channels, the filter only consider how to compute the weighted average of 42 channels. It is difficult to consider the local relationship between neighboring pixels. But if we use 3×3 filters, the network could consider contextual information (compute weighted average of $42 \times 3 \times 3$ neighboring values) and has a good ability to fit actual scenes.

Our framework contains three parallel CNNs whose filters are with local receptive fields of different sizes. We use the same network structures for all columns like [6] but changed 1×1 kernel and the number of filters to slightly increase model's complexity. For each column, the structure is *Conv-ReLU-Pooling-Conv-ReLU-Pooling-Conv-ReLU-Conv*, and we concat all the feature maps from different columns and merge the feature map volume by $1 \times 3 \times 3 \times 42$ filter. Then, we set the neuron corresponding to the area out of ROI to zero based on the ROI mask (images with ROIs are showed in Fig. 1(b)). The ROI mask is optional, because some datasets do not provide ROI information or we do not need to consider ROI in practical problems. And both two pooling layer use 2×2 Max pooling kernel.

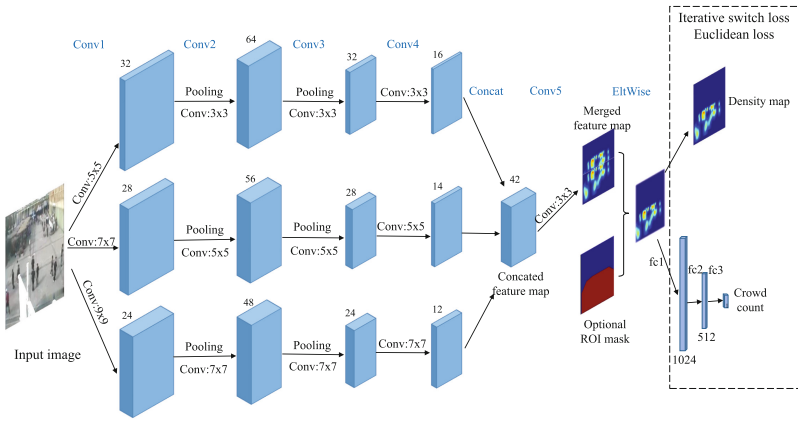


Fig. 2. The structure of the proposed MMCNN model for crowd counting and density map estimation (inherit the structure of MCNN [6]). The text in blue denote manipulations of deep neural network ($Conv_i$ means i th convolutional layer and $Eltwise$ means dot production of two matrix). The cubes in blue denote different feature map volumes after each manipulations. The number above each blue cube is the number of output channels. (Color figure online)

We introduce an iterative switching process in our framework to alternatively optimize the density map estimation task and the count estimation task. The main task is estimating the crowd density map of the input image. Density map prediction needs spatial information and crowd distribution while global count regression only focuses on predicting a numerical value, therefore, using global count as the secondary learning objective can improve the accuracy of our main task. Because of two pooling layers in the CNN model, the output density map is down-sampled to $1/4$ size. Since the density map contains rich and abundant local and detailed information, the CNN model can obtain a better representation of crowd image. The total count regression of the input image is treated as the secondary task and the regression head is directly connected after

Conv5. Then Euclidean distance is used to measure the difference between the estimated density map (global count) and ground truth. The two loss functions are defined as:

$$L_D(\Theta) = \frac{1}{N} \sum_i^N \|F_d(X_i; \Theta) - D_i\|^2, \quad (2)$$

$$L_C(\Theta) = \frac{1}{N} \sum_i^N \|F_c(X_i; \Theta) - C_i\|^2, \quad (3)$$

where Θ is the set of parameters of the CNN model and N is the number of training samples. L_D is the loss between estimated density map $F_d(X_i; \Theta)$ (the output of *Conv5*) and the ground truth density map D_i . Similarly, L_C is the loss between the estimated crowd count $F_c(X_i; \Theta)$ (the output of *fc3*) and the ground truth number C_i .

3.4 Training of MMCNN

The loss functions (2) and (3) can be optimized via batch-based stochastic gradient descent and backpropagation, typical for training neural networks. But in reality, the datasets are very small and the density map data is not balanced (most pixels' value of density map is 0, but the non-zero values are what the loss function really need), it is not easy to learn all the parameters simultaneously. Motivated by [6], we pre-train CNNs in each column separately by our iterative switching process. We then use these pre-trained CNNs to initialize CNNs in all columns and fine-tune all the parameters simultaneously. The iterative switching algorithm we use when we pre-train and fine-tune the CNNs is just like the learning objective switch algorithm in [5]. We firstly use density map to supervise MMCNN's training process due to its abundant spatial information. And then we use global count as training label to promote the precision of our MMCNN model. In this way, our MMCNN model could easily get the optima.

4 Experiment

We evaluate our MMCNN model on The WorldExpo'10 dataset-the largest and most comprehensive dataset in crowd counting field and compare our model to many other methods in the literature. Implementation of the network and its training are based on *Caffe* developed by [35].

Evaluation Metric. By following the convention of existing works [5, 6] for crowd counting, we evaluate different methods with the absolute error (MAE), which is defined as follow:

$$MAE = \frac{1}{N} \sum_1^N |z_i - p(z_i)|, \quad (4)$$

where N is the number of test images, z_i is the actual number of crowd in the i th image, and $p(z_i)$ is the estimated number of crowd in the i th image. In some ways, MAE indicates the accuracy of the estimates.

The WorldExpo'10 Dataset. WorldExpo'10 crowd counting dataset was firstly introduced by [5,37]. This dataset is the largest dataset focusing on crowd counting. It contains 1132 annotated video sequences which are captured by 108 surveillance cameras, all from Shanghai World Expo 2010. The authors of [5] provided a total of 199,923 annotated pedestrians at the centers of their heads in 3980 frames. 3380 frames are used in training data. Testing dataset includes five different video sequences, and each video sequence contains 120 labeled frames. Five different regions of interest (ROI) are provided for the test scenes. Some images are shown in Fig. 1(a).

In this dataset, the perspective maps are given. We apply our Perspective-Adaptive Kernels to obtain density maps. After many times of experiments, we set $C_1 = 0.12$, $C_2 = 0.9$ (details in Sect. 3.1). Because of the noise in the area out of ROI, we only consider the ROI regions. So we use *dot production* to set the neuron corresponding to the area out of ROI to zero. We generate the ROI mask matrix corresponding to each scene. Each mask matrix has the same resolution of density map and the value is either 1 or 0. For evaluation, we use the same evaluation metric (MAE) suggested by [5]. The original input image has 720×560 pixels, we generate density maps have a resolution of 144×180 because our MMCNN model has two 2×2 Max pooling layers. The sum of each density map equals to the total count of pedestrian in the image. The input of our model is original image and RIO mask, output is predicted density map or regressed global count, the whole process is illustrated in Fig. 2.

We use 1/10 of training data as validation data to evaluate the performance and help to optimize hyper parameters. We first split the MMCNN model into three single-column multi-task CNNs, and train each column with two learning tasks iteratively. First of all, we train each column with density map and obtain the optima. Then, we append *fully connected layers* at the end of each CNN and train each CNN with global count, we firstly fix the parameters of convolutional layers until the learning curve come into plateau period. Then, we update all the parameters. In this way, we can obtain the optimal model of each column. We then use these pre-trained single-column CNNs to initialize the columns of MMCNN model and update all the parameters simultaneously in the same way of pre-training each column. For comparing 3×3 kernels and 1×1 kernels, we conducted the experiment with 3×3 kernels in the same method above-mentioned. The result is showed in Fig. 3. The figure tells that 3×3 kernels worked, but multi-task training policy played a greater effect.

In the test phase, the multi-column CNN model is fine-tuned with the first 60 labeled frames for every test scene, and the remaining frames are used as the test data. The results of different methods in the five test video sequences can be seen in Table 1. In scene 2 and scene 5, the density of the crowd changes drastically and there are a lot of noise. Such as scene5, many pedestrians are

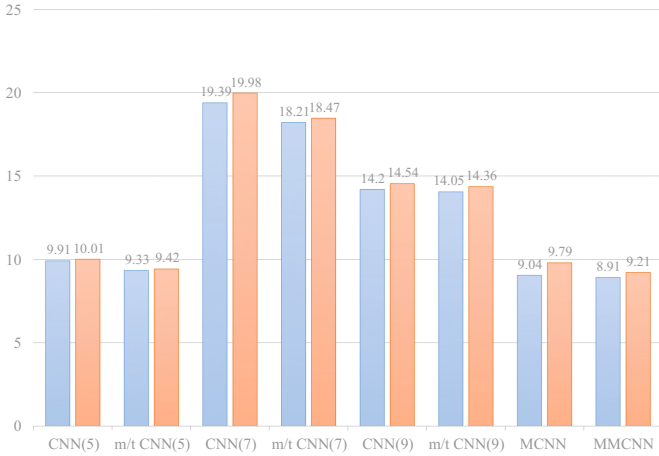


Fig. 3. Comparing single-task CNNs with multi-task CNNs. Blue bars denote test MAE when using 3×3 kernels and orange bars denote test MAE when using 1×1 kernels. $CNN(i)$ means a single column CNN of MMCNN with single task (crowd density map estimation). $m/t CNN(i)$ means single column CNN with multi-task training. $MCNN$ means single-task multi-column CNN. The figure tells that multi-task learning and 3×3 kernels could optimize the performance of our CNN model. And the experiment is conducted on the WorldExpo'10 dataset. (Color figure online)

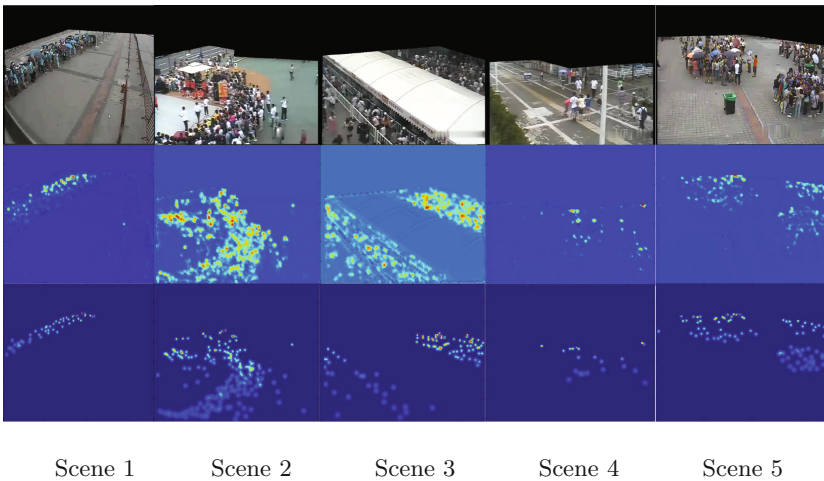


Fig. 4. Test results in each test scene of the WorldExpo'10 dataset. Each column has three pictures which are examples selected from a same scene, the first row are input images with ROIs, the second row are predicted density maps of MMCNN, the third row are the ground truths.

taking umbrella and wearing a hat. In these two scenes, [5] works better. But our model also achieves better performance than Fine-tuned Crowd CNN model [5] in terms of average MAE. Compared to MCNN [6], our model performs better in scene 2 and scene 5 because our 3×3 filter in *Conv5* could integrate local spatial information in a 3×3 receptive fields and our model has a similar average MAE with MCNN [6] (Fig. 4).

Table 1. Mean absolute errors of the WorldExpo'10 crowd counting dataset

Method	Scene1	Scene2	Scene3	Scene4	Scene5	Average
Zhang et al. [5]	9.8	14.1	14.3	22.2	3.7	12.9
MCNN [6]	3.4	20.6	12.9	13.0	8.1	11.6
MMCNN	3.4	18.5	9.6	8.7	7.9	9.6

5 Conclusion

In this paper, we have proposed a Multi-task Multi-column Convolutional Neural Network which can accurately estimate crowd global count and crowd density map in a still image. We evaluate the performance of our model in WorldExpo'10 dataset which is the most representative dataset in crowd counting field. In this dataset, MMCNN outperforms the state-of-art crowd counting methods [6] in each scene and also performs better than Fine-tuned Crowd CNN model [5] in terms of average MAE. Compared to [5], the result reflects that our head-shaped Gaussian kernel is not robust enough in such scene where there are many umbrellas and hats.

Acknowledgement. This work was supported by NSFC (No.71673293).

References

1. Zeng, X., Ouyang, W., Wang, M., Wang, X.: Deep learning of scene-specific classifier for pedestrian detection. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 472–487. Springer, Cham (2014). doi:[10.1007/978-3-319-10578-9_31](https://doi.org/10.1007/978-3-319-10578-9_31)
2. Chen, K., Gong, S., Xiang, T., Mary, Q., Loy, C.C.: Cumulative attribute space for age and crowd density estimation. In: CVPR (2013)
3. Chen, K., Loy, C.C., Gong, S., Xiang, T.: Feature mining for localised crowd counting. In: BMVC (2012)
4. Loy, C.C., Gong, S., Xiang, T.: From semi-supervised to transfer counting of crowds. In: ICCV (2013)
5. Zhang, C., Li, H., Wang, X., Yang, X.: Cross-scene crowd counting via deep convolutional neural networks. In: CVPR (2015)
6. Zhang, Y., Zhou, D., Chen, S., Gao, S., Ma, Y.: Single-image crowd counting via multi-column convolutional neural network. In: CVPR (2016)

7. Viola, P., Jones, M.J., Snow, D.: Detecting pedestrians using patterns of motion and appearance. *Int. J. Comput. Vis.* **63**(2), 153–161 (2005)
8. Lin, Z., Davis, L.S.: Shape-based human detection and segmentation via hierarchical part-template matching. *Pattern Anal. Mach. Intell.* **32**(4), 604–618 (2010)
9. M. Wang and X. Wang. Automatic adaptation of a generic pedestrian detector to a specific traffic scene. In: *CVPR*, pp. 3401–3408. IEEE (2011)
10. Wu, B., Nevatia, R.: Detection of multiple, partially occluded humans in a single image by Bayesian combination of edgelet part detectors. In: *ICCV*, vol. 1, pp. 90–97. IEEE (2005)
11. Idrees, H., Soomro, K., Shah, M.: Detecting humans in dense crowds using locally-consistent scale prior and global occlusion reasoning. *Pattern Anal. Mach. Intell.* **37**(10), 1986–1998 (2005)
12. Zhao, T., Nevatia, R., Wu, B.: Segmentation and tracking of multiple humans in crowded environments. *Pattern Anal. Mach. Intell.* **30**(7), 1198–1211 (2008)
13. Li, M., Zhang, Z., Huang, K., Tan, T.: Estimating the number of people in crowded scenes by mid based foreground segmentation and head-shoulder detection. In: *ICPR*, pp. 1–4. IEEE (2008)
14. Ge, W., Collins, R.T.: Marked point processes for crowd counting. In: *CVPR*, pp. 2913–2920. IEEE (2009)
15. Brostow, G.J., Cipolla, R.: Unsupervised Bayesian detection of independent motion in crowds. In: *CVPR*, vol. 1, pp. 594–601. IEEE (2006)
16. Rabaud, V., Belongie, S.: Counting crowded moving objects. In: *CVPR*, vol. 1, pp. 705–711. IEEE (2006)
17. Chan, A.B., Liang, Z.-S.J., Vasconcelos, N.: Privacy preserving crowd monitoring: counting people without people models or tracking. In: *CVPR*, pp. 1–7. IEEE (2008)
18. Chen, K., Loy, C.C., Gong, S., Xiang, T.: Feature mining for localised crowd counting. In: *BMVC*, vol. 1, p. 3 (2012)
19. Chan, A.B., Vasconcelos, N.: Bayesian poisson regression for crowd counting. In: *ICCV*, pp. 545–551. IEEE (2009)
20. Ryan, D., Denman, S., Fookes, C., Sridharan, S.: Crowd counting using multiple local features. In: *Digital Image Computing: Techniques and Applications*, pp. 81–88. IEEE (2009)
21. Kong, D., Gray, D., Tao, H.: Counting pedestrians in crowds using viewpoint invariant training. In: *BMVC, Citeseer* (2005)
22. Liu, B., Vasconcelos, N.: Bayesian model adaptation for crowd counts. In: *ICCV* (2015)
23. Paragios, N., Ramesh, V.: A MRF-based approach for real-time subway monitoring. In: *CVPR*, vol. 1, p. I-1034. IEEE (2001)
24. Regazzoni, C.S., Tesei, A.: Distributed data fusion for real-time crowding estimation. *Signal Process.* **53**(1), 47–63 (1996)
25. Marana, A., Costa, L.d.F., Lotufo, R., Velastin, S.: On the efficacy of texture analysis for crowd monitoring. In: *International Symposium on Computer Graphics, Image Processing, and Vision*, pp. 354–361. IEEE (1998)
26. Idrees, H., Saleemi, I., Seibert, C., Shah, M.: Multi-source multi-scale counting in extremely dense crowd images. In: *CVPR*, pp. 2547–2554. IEEE (2013)
27. Bansal, A., Venkatesh, K.: People counting in high density crowds from still images. *arXiv preprint arXiv: 1507.08445* 2015
28. Tota, K., Idrees, H.: Counting in dense crowds using deep features
29. Li, W., Zhao, R., Xiao, T., Wang, X.: DeepReID: deep filter pairing neural network for person re-identification. In: *CVPR* (2014)

30. Zeng, X., Ouyang, W., Wang, X.: Multi-stage contextual deep learning for pedestrian detection. In: ICCV (2013)
31. Ouyang, W., Wang, X.: Joint deep learning for pedestrian detection. In: ICCV (2013)
32. Wang, N., Yeung, D.-Y.: Learning a deep compact image representation for visual tracking. In: NIPS (2013)
33. Jing, S., Kai, K., Chang, L.C., Xiaogang, W.: Deeply learned attributes for crowd scene understanding. In: CVPR (2015)
34. Kai, K., Xiaogang, W.: Fully convolutional neural networks for crowd segmentation. arXiv preprint [arXiv:1411.4464](https://arxiv.org/abs/1411.4464) (2014)
35. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: convolutional architecture for fast feature embedding. arXiv preprint [arXiv:1408.5093](https://arxiv.org/abs/1408.5093) (2014)
36. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional models for semantic segmentation. In: CVPR (2015)
37. Zhang, C., Zhang, K., Li, H., Wang, X., Yang, X.: Data-driven crowd understanding: a baseline for a large-scale crowd dataset. *IEEE Trans. Multimedia* **18**(6), 1048–1061 (2016)