Single-Image Crowd Counting via Multi-Column Convolutional Neural Network

Yingying Zhang Desen Zhou Siqin Chen Shenghua Gao Yi Ma
Shanghaitech University
{zhangyy2,zhouds,chensq,gaoshh,mayi}@shanghaitech.edu.cn

Abstract

This paper aims to develop a method than can accurately estimate the crowd count from an individual image with arbitrary crowd density and arbitrary perspective. To this end, we have proposed a simple but effective Multi-column Convolutional Neural Network (MCNN) architecture to map the image to its crowd density map. The proposed MCNN allows the input image to be of arbitrary size or resolution. By utilizing filters with receptive fields of different sizes, the features learned by each column CNN are adaptive to variations in people/head size due to perspective effect or image resolution. Furthermore, the true density map is computed accurately based on geometry-adaptive kernels which do not need knowing the perspective map of the input image. Since exiting crowd counting datasets do not adequately cover all the challenging situations considered in our work, we have collected and labelled a large new dataset that includes 1198 images with about 330,000 heads annotated. On this challenging new dataset, as well as all existing datasets, we conduct extensive experiments to verify the effectiveness of the proposed model and method. In particular, with the proposed simple MCNN model, our method outperforms all existing methods. In addition, experiments show that our model, once trained on one dataset, can be readily transferred to a new dataset.

1. Introduction

In the new year eve of 2015, 35 people were killed in a massive stampede in Shanghai, China. Unfortunately, since then, many more massive stampedes have taken place around the world which have claimed many more victims. Accurately estimating crowds from images or videos has become an increasingly important application of computer vision technology for purposes of crowd control and public safety. In some scenarios, such as public rallies and sports events, the number or density of participating people is an essential piece of information for future event planning and space design. Good methods of crowd counting can also be extended to other domains, for instance, counting cells or bacteria from microscopic images, animal crowd estimates in wildlife sanctuaries, or estimating the number of vehicles at transportation hubs or traffic jams, etc.

Related work. Many algorithms have been proposed in the literature for crowd counting. Earlier methods [29] adopt a detection-style framework that scans a detector over two consecutive frames of a video sequence to estimate the number of pedestrians, based on boosting appearance and motion features. [19, 30, 31] have used a similar detection-based framework for pedestrian counting. In detection-based crowd counting methods, people typically assume a crowd is composed of individual entities which can be detected by some given detectors [13, 34, 18, 10]. The limitation of such detection-based methods is that occlusion among people in a clustered environment or in a very dense crowd significantly affects the performance of the detector hence the final estimation accuracy.

In counting crowds in videos, people have proposed to cluster trajectories of tracked visual features. For instance, [24] has used highly parallelized version of the KLT tracker and agglomerative clustering to estimate the number of moving people. [3] has tracked simple image features and probabilistically group them into clusters representing independently moving entities. However, such tracking-based methods do not work for estimating crowds from individual still images.

Arguably the most extensively used method for crowd counting is feature-based regression, see [4, 7, 5, 27, 15, 20]. The main steps of this kind of method are: 1) segmenting the foreground; 2) extracting various features from the foreground, such as area of crowd mask [4, 7, 27, 23], edge count [4, 7, 27, 25], or texture features [22, 7]; 3) utilizing a regression function to estimate the crowd count. Linear [23] or piece-wise linear [25] functions are relatively simple models and yield decent performance. Other more advanced/effective methods are ridge regression (RR) [7], Gaussian process regression (GPR) [4], and neural network (NN) [22].

There have also been some works focusing on crowd counting from still images. [12] has proposed to leverage

multiple sources of information to compute an estimate of the number of individuals present in an extremely dense crowd visible in a single image. In that work, a dataset of fifty crowd images containing $64K$ annotated humans (UCF_CC_50) is introduced. [2] has followed the work and estimated counts by fusing information from 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.

Recently Zhang *et al*. [33] has proposed a CNN based method to count crowd in different scenes. They first pretrain a network for certain scenes. When a test image from a new scene is given, they choose similar training data to fine-tune the pretrained network based on the perspective information and similarity in density map. Their method demonstrates good performance on most existing datasets. But their method requires perspective maps both on training scenes and the test scene. Unfortunately, in many practical applications of crowd counting, the perspective maps are not readily available, which limits the applicability of such methods.

Contributions of this paper. In this paper, we aim to conduct accurate crowd counting from an arbitrary still image, with an arbitrary camera perspective and crowd density (see Figure 1 for some typical examples). At first sight this seems to be a rather daunting task, since we obviously need to conquer series of challenges:

1. Foreground segmentation is indispensable in most existing work. However foreground segmentation is a challenging task all by itself and inaccurate segmentation will have irreversible bad effect on the final count. In our task, the viewpoint of an image can be arbitrary. Without information about scene geometry or motion, it is almost impossible to segment the crowd from its background accurately. Hence, we have to estimate the number of crowd without segmenting the foreground first.

2. The density and distribution of crowd vary significantly in our task (or datasets) and typically there are tremendous occlusions for most people in each image. Hence traditional detection-based methods do not work well on such images and situations.

3. As there might be significant variation in the scale of the people in the images, we need to utilize features at different scales all together in order to accurately estimate crowd counts for different images. Since we do not have tracked features and it is difficult to handcraft features for all different scales, we have to resort to methods that can automatically learn effective features.

(a)

(b)

Figure 1: (a) Representative images of Part_A in our new crowd dataset. (b) Representative images of Part_B in our crowd dataset. All faces are blurred in (b) for privacy preservation.

To overcome above challenges, in this work, we propose a novel framework based on convolutional neural network (CNN) [9, 16] for crowd counting in an arbitrary still image. More specifically, we propose a multi-column convolutional neural network (MCNN) inspired by the work of [8], which has proposed multi-column deep neural networks for image classification. In their model, an arbitrary number of columns can be trained on inputs preprocessed in different ways. Then final predictions are obtained by averaging individual predictions of all deep neural networks. Our MCNN contains three columns of convolutional neural networks whose filters have different sizes. Input of the MCNN is the image, and its output is a crowd density map whose integral gives the overall crowd count. Contributions of this paper are summarized as follows:

1. The reason for us to adopt a multi-column architecture here is rather natural: the three columns correspond to filters with receptive fields of different sizes (large, medium, small) so that the features learned by each column CNN is adaptive to (hence the overall network is robust to) large variation in people/head size due to perspective effect or across different image resolutions.

2. In our MCNN, we replace the fully connected layer with a convolution layer whose filter size is 1×1. Therefore the input image of our model can be of arbitrary size to avoid distortion. The immediate output of the network is an estimate of the density of the crowd from which we derive the overall count.

3. We collect a new dataset for evaluation of crowd

counting methods. Existing crowd counting datasets cannot fully test the performance of an algorithm in the diverse scenarios considered by this work because their limitations in the variation in viewpoints (UCSD, WorldExpo'10), crowd counts (UCSD), the scale of dataset (UCSD, UCF_CC_50), or the variety of scenes (UCF_CC_50). In this work we introduce a new large-scale crowd dataset named Shanghaitech of nearly 1,200 images with around 330,000 accurately labeled heads. As far as we know, it is the largest crowd counting dataset in terms of number annotated heads. No two images in this dataset are taken from the same viewpoint. This dataset consists of two parts: Part_A and Part_B. Images in Part_A are randomly crawled from the Internet, most of them have a large number of people. Part_B are taken from busy streets of metropolitan areas in Shanghai. We have manually annotated both parts of images and will share this dataset by request. Figure 1 shows some representative samples of this dataset.

2. Multi-column CNN for Crowd Counting

2.1. Density map based crowd counting

To estimate the number of people in a given image via the Convolutional Neural Networks (CNNs), there are two natural configurations. One is a network whose input is the image and the output is the estimated head count. The other one is to output a density map of the crowd (say how many people per square meter), and then obtain the head count by integration. In this paper, we are in favor of the second choice for the following reasons:

1. Density map preserves more information. Compared to the total number of the crowd, density map gives the spatial distribution of the crowd in the given image, and such distribution information is useful in many applications. For example, if the density in a small region is much higher than that in other regions, it may indicate something abnormal happens there.

2. In learning the density map via a CNN, the learned filters are more adapted to heads of different sizes, hence more suitable for arbitrary inputs whose perspective effect varies significantly. Thus the filters are more semantic meaningful, and consequently improves the accuracy of crowd counting.

2.2. Density map via geometry-adaptive kernels

Since the CNN needs to be trained to estimate the crowd density map from an input image, the quality of density given in the training data very much determines the performance of our method. We first describe how to convert an image with labeled people heads to a map of crowd density. If there is a head at pixel $\boldsymbol{x}_i$, we represent it as a delta function $\delta(\boldsymbol{x}-\boldsymbol{x}_i)$. Hence an image with N heads labeled can be represented as a function

$$H(\boldsymbol{x}) = \sum_{i=1}^{N} \delta(\boldsymbol{x}-\boldsymbol{x}_i).$$

To convert this to a continuous density function, we may convolve this function with a Gaussian kernel[17] G_σ so that the density is $F(\boldsymbol{x}) = H(\boldsymbol{x}) * G_\sigma(\boldsymbol{x})$. However, such a density function assumes that these $\boldsymbol{x}_i$ are independent samples in the image plane which is *not* the case here: In fact, each $\boldsymbol{x}_i$ is a sample of the crowd density on the ground in the 3D scene and due to the perspective distortion, and the pixels associated with different samples $\boldsymbol{x}_i$ correspond to areas of different sizes in the scene.

Therefore, to accurately estimate the crowd density F, we need to take into account the distortion caused by the homography between the ground plane and the image plane. Unfortunately, for the task (and datasets) at hand, we typically do not know the geometry of the scene. Nevertheless, if we assume around each head, the crowd is somewhat evenly distributed, then the average distance between the head and its nearest k neighbors (in the image) gives a reasonable estimate of the geometric distortion (caused by the perspective effect).

Therefore, we should determine the spread parameter σ based on the size of the head for each person within the image. However, in practice, it is almost impossible to accurately get the size of head due to the occlusion in many cases, and it is also difficult to find the underlying relationship between the head size the density map. Interesting we found that usually the head size is related to the distance between the centers of two neighboring persons in crowded scenes (please refer to Figure 2). As a compromise, for the density maps of those crowded scenes, we propose to data-adaptively determine the spread parameter for each person based on its average distance to its neighbors.[1]

For each head $\boldsymbol{x}_i$ in a given image, we denote the distances to its k nearest neighbors as $\{d_1^i, d_2^i, \ldots, d_m^i\}$. The average distance is therefore $\bar{d}^i = \frac{1}{m}\sum_{j=1}^{m} d_j^i$. Thus, the pixel associated with $\boldsymbol{x}_i$ corresponds to an area on the ground in the scene roughly of a radius proportional to $\bar{d}^i$. Therefore, to estimate the crowd density around the pixel $\boldsymbol{x}_i$, we need to convolve $\delta(\boldsymbol{x}-\boldsymbol{x}_i)$ with a Gaussian kernel with variance σ_i proportional to $\bar{d}^i$, More precisely, the

[1]For the images given the density or perspective maps, we directly use the given density maps in our experiments or use the density maps generated from perspective maps. For those data only contain very few persons and the sizes of heads are similar, we use the fixed spread parameter for all the persons.

density F should be

$$F(\boldsymbol{x}) = \sum_{i=1}^{N} \delta(\boldsymbol{x} - \boldsymbol{x}_i) * G_{\sigma_i}(\boldsymbol{x}), \quad \text{with} \quad \sigma_i = \beta \bar{d}^i$$

for some parameter β. In other words, we convolve the labels H with density kernels adaptive to the local geometry around each data point, referred to as geometry-adaptive kernels. In our experiment, we have found empirically $\beta = 0.3$ gives the best result. In Figure 2, we have shown so-obtained density maps of two exemplar images in our dataset.

Figure 2: Original images and corresponding crowd density maps obtained by convolving geometry-adaptive Gaussian kernels.

2.3. Multi-column CNN for density map estimation

Due to perspective distortion, the images usually contain heads of very different sizes, hence filters with receptive fields of the same size are unlikely to capture characteristics of crowd density at different scales. Therefore, it is more natural to use filters with different sizes of local receptive field to learn the map from the raw pixels to the density maps. Motivated by the success of Multi-column Deep Neural Networks (MDNNs) [8], we propose to use a Multi-column CNN (MCNN) to learn the target density maps. In our MCNN, for each column, we use the filters of different sizes to model the density maps corresponding to heads of different scales. For instance, filters with larger receptive fields are more useful for modeling the density maps corresponding to larger heads.

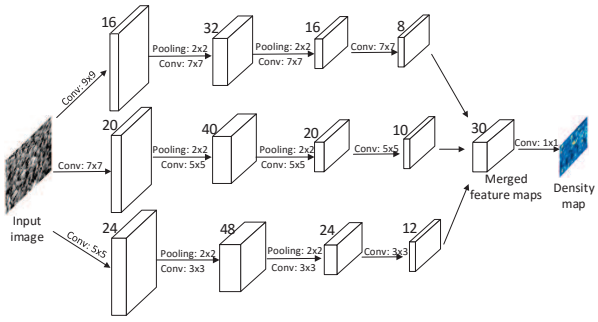


Figure 3: The structure of the proposed multi-column convolutional neural network for crowd density map estimation.

The overall structure of our MCNN is illustrated in Figure 3. It contains three parallel CNNs whose filters are with local receptive fields of different sizes. For simplification, we use the same network structures for all columns (i.e., conv–pooling–conv–pooling) except for the sizes and numbers of filters. Max pooling is applied for each 2×2 region, and Rectified linear unit (ReLU) is adopted as the activation function because of its good performance for CNNs [32]. To reduce the computational complexity (the number of parameters to be optimized), we use less number of filters for CNNs with larger filters. We stack the output feature maps of all CNNs and map them to a density map. To map the features maps to the density map, we adopt filters whose sizes are 1×1 [21]. Then Euclidean distance is used to measure the difference between the estimated density map and ground truth. The loss function is defined as follows:

$$L(\Theta) = \frac{1}{2N} \sum_{i=1}^{N} \|F(X_i; \Theta) - F_i\|_2^2, \tag{1}$$

where Θ is a set of learnable parameters in the MCNN. N is the number of training image. X_i is the input image and F_i is the ground truth density map of image X_i. $F(X_i; \Theta)$ stands for the estimated density map generated by MCNN which is parameterized with Θ for sample X_i. L is the loss between estimated density map and the ground truth density map.

Remarks i) Since we use two layers of max pooling, the spatial resolution is reduced by $\frac{1}{4}$ for each image. So in the training stage, we also down-sample each training sample by $\frac{1}{4}$ before generating its density map. ii) Conventional CNNs usually normalize their input images to the same size. Here we prefer the input images to be of their original sizes because resizing images to the same size will introduce additional distortion in the density map that is difficult to estimate. iii) Besides the fact that the filters have different sizes in our CNNs, another difference between our MCNN and conventional MDNNs is that we combine the outputs of all CNNs with learnable weights (i.e.,1×1 filters). In contrast, in MDNNs proposed by [8], the outputs are simply averaged.

2.4. Optimization of MCNN

The loss function (1) can be optimized via batch-based stochastic gradient descent and backpropagation, typical for training neural networks. However, in reality, as the number of training samples are very limited, and the effect of gradient vanishing for deep neural networks, it is not easy to learn all the parameters simultaneously. Motivated by the success of pre-training of RBM [11], we pre-train CNN in each single column separately by directly mapping the outputs of the fourth convolutional layer to the density map. We then use these pre-trained CNNs to initialize CNNs in all columns and fine-tune all the parameters simultaneously.

2.5. Transfer learning setting

One advantage of such a MCNN model for density estimation is that the filters are learned to model the density maps of heads with different sizes. Thus if the model is trained on a large dataset which contains heads of very different sizes, then the model can be easily adapted (or transferred) to another dataset whose crowd heads are of some particular sizes. If the target domain only contains a few training samples, we may simply fix the first several layers in each column in our MCNN, and only fine-tune the last few convolutional layers. There are two advantages for fine-tuning the last few layers in this case. Firstly, by fixing the first several layers, the knowledge learnt in the source domain can be preserved, and by fine-tuning the last few layers, the models can be adapted to the target domain. So the knowledge in both source domain and target domain can be integrated and help improve the accuracy. Secondly, comparing with fine-tuning the whole network, fine-tuning the last few layers greatly reduces the computational complexity.

3. Experiments

We evaluate our MCNN model on four different datasets – three existing datasets and our own dataset. Although comparing to most DNN based methods in the literature, the proposed MCNN model is not particularly deep nor sophisticated, it has nevertheless achieved competitive and often superior performance in all the datasets. In the end, we also demonstrate the generalizability of such a simple model in the transfer learning setting (as mentioned in section 2.5). Implementation of the proposed network and its training are based on the Caffe framework developed by [14].

3.1. Evaluation metric

By following the convention of existing works [28][33] for crowd counting, we evaluate different methods with both the absolute error (MAE) and the mean squared error (MSE), which are defined as follows:

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

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

3.2. Shanghaitech dataset

As exiting datasets are not entirely suitable for evaluation of the crowd count task considered in this work, we introduce a new large-scale crowd counting dataset named Shanghaitech which contains 1198 annotated images, with a total of 330,165 people with centers of their heads annotated. As far as we know, this dataset is the largest one in terms of the number of annotated people. This dataset consists of two parts: there are 482 images in Part_A which are randomly crawled from the Internet, and 716 images in Part_B which are taken from the busy streets of metropolitan areas in Shanghai. The crowd density varies significantly between the two subsets, making accurate estimation of the crowd more challenging than most existing datasets. Both Part_A and Part_B are divided into training and testing: 300 images of Part_A are used for training and the remaining 182 images for testing;, and 400 images of Part_B are for training and 316 for testing. Table1 gives the statistics of Shanghaitech dataset and its comparison with other datasets. We also give the crowd histograms of images in this dataset in Figure 4. If the work is accepted for publication, we will release the dataset, the annotations, as well as the training/testing protocol.

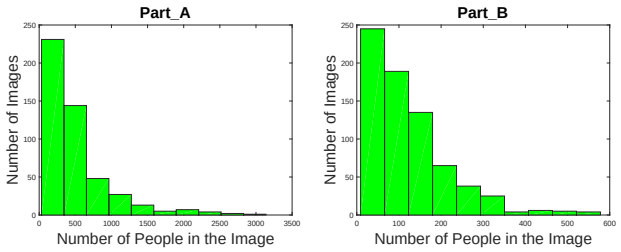


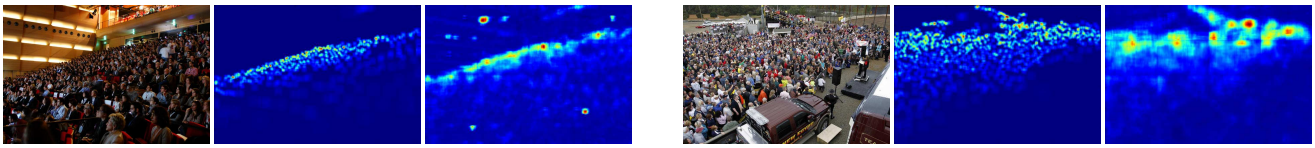
Figure 4: Histograms of crowd counts of our new dataset.

To augment the training set for training the MCNN, we cropped 9 patches from each image at different locations, and each patch is $1/4$ size of the original image. All the patches are used to train our MCNN model. For Part_A, as the crowd density is usually very high, we use our geometry-adaptive kernels to generate the density maps, and the predicted density at overlapping region is calculated by averaging. For Part_B, since the crowd is relatively sparse, we use the same spread in Gaussian kernel to generate the (ground truth) density maps. In our implementation, we first pre-train each column of MCNN independently. Then we fine-tune the whole network. Figure 5 shows examples of ground truth density maps and estimated density maps of images in Part_A.

We compare our method with the work of Zhang *et al*. [33], which also uses CNNs for crowd counting and achieved state-of-the-art accuracy at the time. Following the work of [33], we also compare our work with regression based method, which uses Local Binary Pattern (LBP) features extracted from the original image as input and uses ridge regression (RR) to predict the crowd number for each image. To extract LBP features, each image is uniformly

Table 1: Comparation of Shanghaitech dataset with existing datasets: **Num** is the number of images; **Max** is the maximual crowd count; **Min** is the minimal crowd count; **Ave** is the average crowd count; **Total** is total number of labeled people.

Dataset		Resolution	Num	Max	Min	Ave	Total
UCSD		158×238	2000	46	11	24.9	49,885
UCF_CC_50		different	50	4543	94	1279.5	63,974
WorldExpo		576×720	3980	253	1	50.2	199,923
Shanghaitech	Part_A	different	482	3139	33	501.4	241,677
	Part_B	768×1024	716	578	9	123.6	88,488



Test image Ground-truth Estimation Test image Ground-truth Estimation

Figure 5: The ground truth density map and estimated density map of our MCNN Model of two test images in part_A

divided into 8×8 blocks in Part A and 12×16 blocks in Part B, then a 59-dimensional uniform LBP in each block is extracted and all uniform LBP features are concatenated together to represent the image. The ground truth is a 64D or 192D vector where each entry is the total number of persons in corresponding patch. We compare the performances of all the methods on Shanghaitech dataset in Table 2.

The effect of pretraining in MCNN. We show the effect of our model without pretraining on Shanghaitech dataset Part_A in Figure 6. We see that pretrained network outperforms the network without pretraining. The result verifies the necessity of pretraining for MCNN as optimization starting from random initialization tends to fall into local minima.

Single column CNNs vs MCNN. Figure 6 shows the comparison of single column CNNs with MCNN on Shanghaitech dataset Part_A. It can be seen that MCNNs significantly outperforms each single column CNN for both MAE and MSE. This verifies the effectiveness of the MCNN architecture.

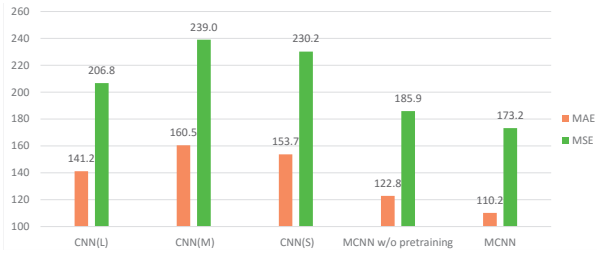


Figure 6: Comparing single column CNNs with MCNN and MCNN w/o pretraining on Part_A. L, M, S stand for large kernel, medium kernel, small kernel respectively.

Comparison of different loss functions. We evaluate the performance of our framework with different loss functions. Other than mapping the images to their density maps, we can also map the images to the total head counts in the image directly. For the input image X_i $(i = 1, \ldots, N)$, its total head count is z_i, and $F(X_i; \Theta)$ stands for the estimated density map and Θ is the parameters of MCNN. Then we arrive the following objective function:

$$L(\Theta) = \frac{1}{2N} \sum_{i=1}^{N} \left\| \iint_S F(X_i; \Theta) dx dy - z_i \right\|^2 \quad (3)$$

Here S stands for the spatial region of estimated density map, and ground truth of the density map is not used. For this loss, we also pretrain CNNs in each column separately. We call such a baseline as MCNN based crowd count regression (MCNN-CCR). Performance based on such loss function is listed in Table 2, which is also compared with two existing methods as well as the method based on density map estimation (simply labeled as MCNN). We see that the results based on crowd count regression is rather poor. In a way, learning density map manages to preserve more information of the image, and subsequently helps improve the count accuracy.

In Figure 7, we compare the results of our method with those of Zhang *et al*. [33] in more details. We group the test images in Part_A and Part_B into 10 groups according to crowd counts in an increasing order. We have 182+316 test images in Part_A and Part_B. Except for the 10th group which contains 20+37 images, other groups all have 18+31 images each. From the plots in the figure, we can see that our method is much more accurate and robust to large variation in crowd number/density.

Table 2: Comparing performances of different methods on Shanghaitech dataset.

	Part_A		Part_B	
Method	**MAE**	**MSE**	**MAE**	**MSE**
LBP+RR	303.2	371.0	59.1	81.7
Zhang *et al*. [33]	181.8	277.7	32.0	49.8
MCNN-CCR	245.0	336.1	70.9	95.9
MCNN	**110.2**	**173.2**	**26.4**	**41.3**

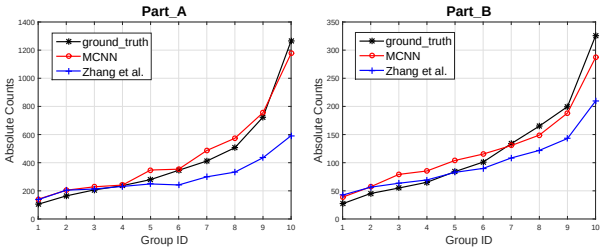


Figure 7: Comparison of our method to Zhang *et al*. [33] on Shanghaitech dataset: We evenly divided our test images into 10 groups according to increasing number of people. Absolute count in the vertical axis is the average crowd number of images in each group.

3.3. The UCF_CC_50 dataset

The UCF_CC_50 dataset is firstly introduced by H. Idrees *et al*. [12]. This dataset contains 50 images from the Internet. It is a very challenging dataset, because of not only limited number of images, but also the crowd count of the image changes dramatically. The head counts range between 94 and 4543 with an average of 1280 individuals per image. The authors provided 63974 annotations in total for these fifty images. We perform 5-fold cross-validation by following the standard setting in [12]. The same data augmentation approach as in that in Shanghaitech dataset.

Table 3: Comparing results of different methods on the UCF_CC_50 dataset.

Method	**MAE**	**MSE**
Rodriguez *et al*. [26]	655.7	697.8
Lempitsky *et al*. [17]	493.4	**487.1**
Idrees *et al*. [12]	419.5	541.6
Zhang *et al*. [33]	467.0	498.5
MCNN	**377.6**	509.1

We compare our method with four existing methods on UCF_CC_50 dataset in Table 3. Rodriguez *et al*. [26] employs density map estimation to obtain better head detection results in crowd scenes. Lempitsky *et al*. [17] adopts dense SIFT features on randomly selected patches and the MESA distance to learn a density regression model. The method presented in [12] gets the crowd count estimation by using multi-source features. The work of Zhang *et al*. [33] is based on crowd CNN model to estimate the crowd count of an image. Our method achieves the best MAE, and comparable MSE with existing methods.

3.4. The UCSD dataset

We also evaluate our method on the UCSD dataset [4]. This dataset contains 2000 frames chosen from one surveillance camera in the UCSD campus. The frame size is 158×238 and it is recoded at 10 fps. There are only about 25 persons on average in each frame (Please refer to Table 1) The dataset provides the ROI for each video frame.

By following the same setting with [4], we use frames from 601 to 1400 as training data, and the remaining 1200 frames are used as test data. This dataset does not satisfy assumptions that the crowd is evenly distributed. So we fix the σ of the density map. The intensities of pixels out of ROI is set to zero, and we also use ROI to revise the last convolution layer. Table 4 shows the results of our method and other methods on this dataset. The proposed MCNN model outperforms both the foreground segmentation based methods and CNN based method [33]. This indicates that our model can estimate not only images with extremely dense crowds but also images with relative sparse people.

Table 4: Comparing results of different methods on the UCSD dataset.

Method	**MAE**	**MSE**
Kernel Ridge Regression [1]	2.16	7.45
Ridge Regression [7]	2.25	7.82
Gaussian Process Regression [4]	2.24	7.97
Cumulative Attribute Regression [6]	2.07	6.86
Zhang *et al*. [33]	1.60	3.31
MCNN	**1.07**	**1.35**

3.5. The WorldExpo'10 dataset

WorldExpo'10 crowd counting dataset was firstly introduced by Zhang *et al*. [33]. This dataset contains 1132 annotated video sequences which are captured by 108 surveillance cameras, all from Shanghai 2010 WorldExpo. The authors of [33] 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.

In this dataset, the perspective maps are given. For fair

Table 5: Mean absolute errors of the WorldExpo'10 crowd counting dataset.

Method	Sence1	Sence2	Sence3	Sence4	Sence5	Average
LBP + RR	13.6	59.8	37.1	21.8	23.4	31.0
Zhang *et al*. [33]	9.8	14.1	14.3	22.2	3.7	12.9
MCNN	3.4	20.6	12.9	13.0	8.1	**11.6**

comparison, we followed the work of [33], generated the density map according to perspective map with the relation $\sigma = 0.2 * M(x)$, $M(x)$ denotes that the number of pixels in the image representing one square meter at that location. To be consistent with [33], only ROI regions are considered in each test scene. So we modify the last convolution layer based on the ROI mask, namely, setting the neuron corresponding to the area out of ROI to zero. We use the same evaluation metric (MAE) suggested by the author of [33]. Table 5 reports the results of different methods in the five test video sequences. Our method also achieves better performance than Fine-tuned Crowd CNN model [33] in terms of average MAE.

3.6. Evaluation on transfer learning

To demonstrate the generalizability of the learned model in our method, we test our method in the transfer learning setting by using the Part_A of Shanghaitech dataset as the source domain and using the UCF_CC_50 dataset as the target domain. Specifically, we train a MCNNs model with data in the source domain. For the crowd counting task in the target domain, we conduct two settings, i.e., (i) no training samples in the target domain, and (ii) There are only a few samples in the target domain. For case (i), we directly use our model trained on Part_A of Shanghaitech dataset for evaluation. For case (ii), we use the training samples in the target domain to fine-tune the network. The performance of different settings is reported in Table 6. The accuracy differences between models trained on UCF_CC_50 and Part_A are similar (377.7 vs 397.7), which means the model trained on Part_A is already good enough for the task on UCF_CC_50. By fine-tuning the last two layers of MCNN with training data on UCF_CC_50, the accuracy can be greatly boosted (377.7 vs. 295.1). However, if the whole network is fine-tuned rather than only the last two layers, the performance drops significantly (295.1 vs 378.3), but still comparable (377.7 vs 378.31) with the MCNN model trained with the training data of the target domain. The performance gap between fine-tuning the whole network and fine-tuning the last couple of layers is perhaps due to the reason that we have limited training samples in the UCF_CC_50 dataset. Fine-tuning the last two layers ensures that the output of the model is adapted to the target domain, and keeping the first few layers of the model intact ensures that good features/filters learned from adequate data in the source domain will be preserved. But if the whole network is fine-tuned with inadequate data in the target domain, the learned model becomes similar to that learned with only the training data in the target domain. Hence the performance degrades to that of the model learned in the latter case.

Table 6: Transfer learning across datasets. "MCNN w/o transfer" means we train the MCNN using the training data in UCF_CC_50 only, and data from the source domain are not used. "MCNN trained on Part_A" means we do not use the training data in the target domain to fine-tune the MCNN trained in the source domain.

Method	**MAE**	**MSE**
MCNN w/o transfer	377.7	509.1
MCNN trained on Part_A	397.7	624.1
Finetune the whole MCNN	378.3	594.6
Finetune the last two layers	**295.1**	**490.23**

4. Conclusion

In this paper, we have proposed a Multi-column Convolution Neural Network which can estimate crowd number accurately in a single image from almost any perspective. To better evaluate performances of crowd counting methods under practical conditions, we have collected and labelled a new dataset named Shanghaitech which consists of two parts with a total of 330,165 people annotated. This is the largest dataset so far in terms of the annotated heads for crowd counting. Our model outperforms the state-of-art crowd counting methods on all datasets used for evaluation. Further, our model trained on a source domain can be easily transferred to a target domain by fine-tuning only the last few layers of the trained model, which demonstrates good generalizability of the proposed model.

5. Acknowledgement

This work was supported by the Shanghai Pujiang Talent Program(No.15PJ1405700), and NSFC (No. 61502304).

References

[1] S. An, W. Liu, and S. Venkatesh. Face recognition using kernel ridge regression. In *CVPR*, pages 1–7. IEEE, 2007.

[2] A. Bansal and K. Venkatesh. People counting in high density crowds from still images. *arXiv preprint arXiv:1507.08445*, 2015.

[3] G. J. Brostow and R. Cipolla. Unsupervised bayesian detection of independent motion in crowds. In *CVPR*, volume 1, pages 594–601. IEEE, 2006.

[4] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *CVPR*, pages 1–7. IEEE, 2008.

[5] A. B. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting. In *ICCV*, pages 545–551. IEEE, 2009.

[6] K. Chen, S. Gong, T. Xiang, and C. C. Loy. Cumulative attribute space for age and crowd density estimation. In *CVPR*, pages 2467–2474. IEEE, 2013.

[7] K. Chen, C. C. Loy, S. Gong, and T. Xiang. Feature mining for localised crowd counting. In *BMVC*, volume 1, page 3, 2012.

[8] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *CVPR*, pages 3642–3649. IEEE, 2012.

[9] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980.

[10] W. Ge and R. T. Collins. Marked point processes for crowd counting. In *CVPR*, pages 2913–2920. IEEE, 2009.

[11] G. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *NEURAL COMPUT*, 18(7):1527–1554, 2006.

[12] H. Idrees, I. Saleemi, C. Seibert, and M. Shah. Multi-source multi-scale counting in extremely dense crowd images. In *CVPR*, pages 2547–2554. IEEE, 2013.

[13] H. Idrees, K. Soomro, and M. Shah. Detecting humans in dense crowds using locally-consistent scale prior and global occlusion reasoning. *Pattern Analysis and Machine Intelligence*, 2005.

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[15] D. Kong, D. Gray, and H. Tao. Counting pedestrians in crowds using viewpoint invariant training. In *BMVC*. Citeseer, 2005.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[17] V. Lempitsky and A. Zisserman. Learning to count objects in images. In *Advances in Neural Information Processing Systems*, pages 1324–1332, 2010.

[18] M. Li, Z. Zhang, K. Huang, and T. Tan. Estimating the number of people in crowded scenes by mid based foreground segmentation and head-shoulder detection. In *ICPR*, pages 1–4. IEEE, 2008.

[19] Z. Lin and L. S. Davis. Shape-based human detection and segmentation via hierarchical part-template matching. *Pattern Analysis and Machine Intelligence*, 32(4):604–618, 2010.

[20] B. Liu and N. Vasconcelos. Bayesian model adaptation for crowd counts. In *ICCV*, 2015.

[21] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1411.4038*, 2014.

[22] A. Marana, L. d. F. Costa, R. Lotufo, and S. Velastin. On the efficacy of texture analysis for crowd monitoring. In *International Symposium on Computer Graphics, Image Processing, and Vision*, pages 354–361. IEEE, 1998.

[23] N. Paragios and V. Ramesh. A mrf-based approach for real-time subway monitoring. In *CVPR*, volume 1, pages I–1034. IEEE, 2001.

[24] V. Rabaud and S. Belongie. Counting crowded moving objects. In *CVPR*, volume 1, pages 705–711. IEEE, 2006.

[25] C. S. Regazzoni and A. Tesei. Distributed data fusion for real-time crowding estimation. *Signal Processing*, 53(1):47–63, 1996.

[26] M. Rodriguez, I. Laptev, J. Sivic, and J.-Y. Audibert. Density-aware person detection and tracking in crowds. In *ICCV*, pages 2423–2430. IEEE, 2011.

[27] D. Ryan, S. Denman, C. Fookes, and S. Sridharan. Crowd counting using multiple local features. In *Digital Image Computing: Techniques and Applications*, pages 81–88. IEEE, 2009.

[28] K. Tota and H. Idrees. Counting in dense crowds using deep features.

[29] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2):153–161, 2005.

[30] M. Wang and X. Wang. Automatic adaptation of a generic pedestrian detector to a specific traffic scene. In *CVPR*, pages 3401–3408. IEEE, 2011.

[31] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *ICCV*, volume 1, pages 90–97. IEEE, 2005.

[32] M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, and J. Dean. On rectified linear units for speech processing. In *ICASSP*, pages 3517–3521. IEEE, 2013.

[33] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-scene crowd counting via deep convolutional neural networks. In *CVPR*, 2015.

[34] T. Zhao, R. Nevatia, and B. Wu. Segmentation and tracking of multiple humans in crowded environments. *Pattern Analysis and Machine Intelligence*, 30(7):1198–1211, 2008.