Revisiting Perspective Information for Efficient Crowd Counting

Miaojing Shi[1], Zhaohui Yang[2], Chao Xu[2], Qijun Chen[3]
[1]Univ Rennes, Inria, CNRS, IRISA
[2]Key Laboratory of Machine Perception, Cooperative Medianet Innovation Center, Peking University
[3]Department of Control Science and Engineering, Tongji University

Abstract

Crowd counting is the task of estimating people numbers in crowd images. Modern crowd counting methods employ deep neural networks to estimate crowd counts via crowd density regressions. A major challenge of this task lies in the perspective distortion, which results in drastic person scale change in an image. Density regression on the small person area is in general very hard. In this work, we propose a perspective-aware convolutional neural network (PACNN) for efficient crowd counting, which integrates the perspective information into density regression to provide additional knowledge of the person scale change in an image. Ground truth perspective maps are firstly generated for training; PACNN is then specifically designed to predict multi-scale perspective maps, and encode them as perspective-aware weighting layers in the network to adaptively combine the outputs of multi-scale density maps. The weights are learned at every pixel of the maps such that the final density combination is robust to the perspective distortion. We conduct extensive experiments on the ShanghaiTech, WorldExpo'10, UCF_CC_50, and UCSD datasets, and demonstrate the effectiveness and efficiency of PACNN over the state-of-the-art.

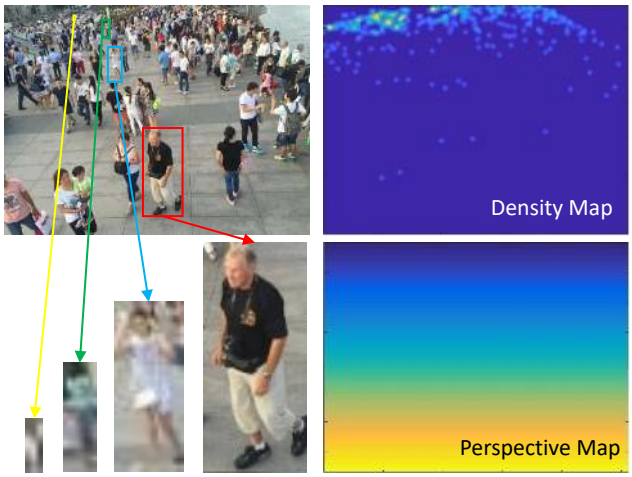

Figure 1: The density map shows the locally smoothed crowd count at every location in the image. The perspective map reflects the perspective distortion at every location in the image, e.g. how many pixels correspond to a human height of one meter at each location [46]. Person scale changes drastically due to the perspective distortion. Density regression on the small person area is in general very hard. We integrate the perspective map into density regression to provide additional information about the general person scale change from near to far in the image.

1. Introduction

The rapid growth of the world's population has led to fast urbanization and resulted in more frequent crowd gatherings, e.g. sport events, music festivals, political rallies. Accurate and fast crowd counting thereby becomes essential to handle large crowds for public safety. Traditional crowd counting methods estimate crowd counts via the detection of each individual pedestrian [44, 39, 3, 27, 21]. Recent methods conduct crowd counting via the regression of density maps [5, 7, 30, 12]: the problem of crowd counting is casted as estimating a continuous density function whose integral over an image gives the count of persons within that image [7, 15, 16, 25, 46, 47, 31] (see Fig. 1: Density Map). Handcrafted features were firstly employed in the density regression [7, 15, 16] and soon outperformed by deep representations [25, 46, 47].

A major challenge of this task lies in the drastic perspective distortions in crowd images (see Fig. 1). The perspective problem is related to camera calibration which estimates a camera's 6 degrees-of freedom (DOF) [10]. Besides the camera DOFs, it is also defined in way to signify the person scale change from near to far in an image in crowd counting task [5, 6, 46, 11]. Perspective information has been widely used in traditional crowd counting methods to normalize features extracted at different locations of the image [5, 16, 9, 22]. Despite the great benefits achieved by using image perspectives, there exists one clear disadvantage regarding its acquisition, which normally requires additional information/annotations of the camera parameters

or scene geometries. The situation becomes serious when the community starts to employ deep learning to solve the problem in various scenarios [47, 12], where the perspective information is usually unavailable or not easy to acquire. While some works propose certain simple ways to label the perspective maps [5, 46], most researchers in recent trends work towards a perspective-free setting [25] where they exploit the multi-scale architecture of convolutional neural networks (CNNs) to regress the density maps at different resolutions [47, 40, 25, 36, 31, 28, 4]. To account for the varying person scale and crowd density, the patch-based estimation scheme [46, 25, 36, 31, 45, 19, 32, 4] is usually adopted such that different patches are predicted (inferred) with different contexts/scales in the network. The improvements are significant but time costs are also expensive.

In this work, we revisit the perspective information for efficient crowd counting. We show that, with a little effort on the perspective acquisition, we are able to generate perspective maps for varying density crowds. We propose to integrate the perspective maps into crowd density regression to provide additional information about the person scale change in an image, which is particularly helpful on the density regression of small person area. The integration directly operates on the pixel-level, such that the proposed approach can be both efficient and accurate. To summarize, we propose a perspective-aware CNN (PACNN) for crowd counting. The contribution of our work concerns two aspects regarding the perspective generation and its integration with crowd density regression:

(A) The ground truth perspective maps are firstly generated for network training: sampled perspectives are computed at several person locations based on their relations to person size; a specific nonlinear function is proposed to fit the sampled values in each image based on the perspective geometry. Having the ground truth, we train the network to directly predict perspective maps for new images.

(B) The perspective maps are explicitly integrated into the network to guide the multi-scale density combination: three outputs are adaptively combined via two perspective-aware weighting layers in the network , where the weights in each layer are learned through a nonlinear transform of the predicted perspective map at the corresponding resolution. The final output is robust to the perspective distortion; we thereby infer the crowd density over the entire image.

We conduct extensive experiments on several standard benchmarks i.e. ShanghaiTech [47], WorldExpo'10 [46], UCF_FF_50 [12] and UCSD [5], to show the superiority of our PACNN over the state-of-the-art.

2. Related work

We categorize the literature in crowd counting into traditional and modern methods. Modern methods refer to those employ CNNs while traditional methods do not.

2.1. Traditional methods

Detection-based methods. These methods consider a crowd as a group of detected individual pedestrians [21, 42, 44, 37, 39, 3, 27]. They can be performed either in a monolithic manner or part-based. Monolithic approach typically refers to pedestrian detection that employs handcrafted features like Haar [38] and HOG [8] to train an SVM or AdaBoost detector [37, 39, 3, 27]. These approaches often perform poorly in the dense crowds where pedestrians are heavily occluded or overlapped. Part-based detection is therefore adopted in many works [18, 42, 44, 13] to count pedestrian from parts in images. Despite the improvements achieved, the detection-based crowd counting overall suffers severely in dense crowds with complex backgrounds.

Regression-based methods. These methods basically have two steps: first, extracting effective features from crowd images; second, utilizing various regression functions to estimate crowd counts. Regression features include edge features [5, 7, 30, 29, 6], texture features [7, 12, 24, 6] etc. Regression methods include linear [29, 26], ridge [7] and Gaussian [5, 6] functions. Earlier works ignore the spatial information by simply regressing a scalar value (crowd count), later works instead learn a mapping from local features to a density map [7, 15, 16]. Spatial locations of persons are encoded into the density map; the crowd count is obtained by integrating over the density map.

Perspective information was widely used in traditional crowd counting methods, which provides additional information regarding the person scale change along with the perspective geometry. It is usually utilized to normalize the regression features or detection results [5, 16, 22, 13].

2.2. Modern methods

Due to the use of strong CNN features, recent works on crowd counting have shown remarkable progress [46, 2, 47, 25, 48, 35, 36, 31, 45, 23, 19, 20, 17, 33, 32, 4, 28]. In order to deal with the varying head size in one image, the multi-column [47, 25, 31, 2] or multi-scale [23, 4, 32, 28] network architecture is often utilized for crowd density regression. Many works also adopt a patch-based scheme to divide each image into local patches corresponding to different crowd densities and scales [25, 31, 32, 4]. For example, [25] uses a pyramid of image patches extracted at multiple scales and feeds them into different CNN columns; while Sam et al. [31] introduce a switch classifier to relay the crowd patches from images to their best CNN columns with most suitable scales. Sindagi et al. [36] design a system called contextual pyramid CNN. It consists of both a local and global context estimator to perform patch-based density estimation.Shen et al. [32] introduce an adversarial loss to generate density map for sharper and higher resolution and design a novel scale-consistency regularizer which enforces that the sum of the crowd counts from local patches is co-

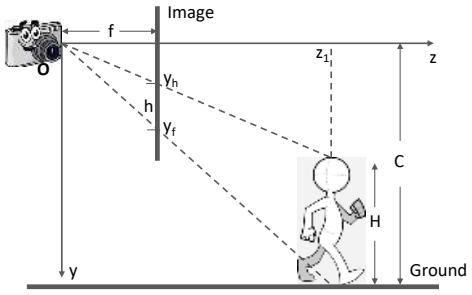

Figure 2: The perspective geometry of a pinhole camera seen from the x-axis. The Cartesian coordinate system starts from origin **O**, with y-axis representing the vertical direction while z-axis the optical axis (depth). A person with true height H is walking on the ground, and he is shot by a camera located at **O** where the camera aperture is. The person's head top and feet bottom are mapped on the image plane at y_h and y_f, respectively. The distance from the camera aperture to the image plane is f, which is also known as the focal length. The camera height from the ground is C.

herent with the overall count of their region union. Cao et al. [4] propose a novel encoder-decoder network and local pattern consistency loss in crowd counting. A patch-based test scheme is also applied to reduce the impact of statistic shift problem.

Perspective information was also used in modern methods but often in an implicit way, e.g. to normalize the scales of pedestrians in the generalization of ground truth density [46, 47] or body part [11] maps. We instead predict the perspective maps directly in the network and use them to adaptively combine the multi-scale density outputs. There are also other works trying to learn or leverage different cues to address the perspective distortion in images [13, 1]. For instance, [13] uses locally-consistent scale prior maps to detect and count humans in dense crowds; while [1] employs a depth map to predict the size of objects in the wild and count them.

3. Perspective-aware CNN

In this section we first generate ground truth density maps and perspective maps; then introduce the network architecture; finally present the network training protocol.

3.1. Ground truth (GT) generation

GT density map generation. The GT density map D^g can be generated by convolving Gaussian kernel G_σ with head center annotation z_j, as in [47, 31, 36]:

$$D^g = \sum_{j=1}^{Y^g} G_\sigma(z - z_j), \tag{1}$$

where Y^g denotes the total number of persons in an image; σ is obtained following [47]. The integral of D^g is equivalent to Y^g (see Fig. 1).

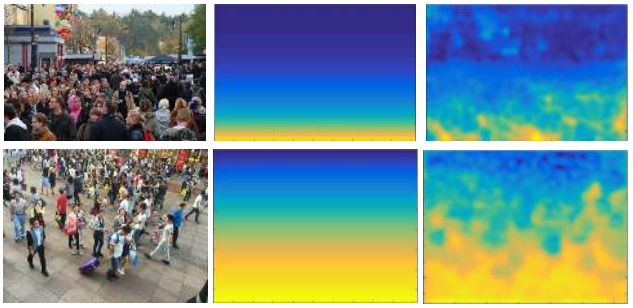

Figure 3: Perspective samples from SHA and SHB [47]. In each row, the left column is the original image, middle column is the GT perspective map using (6) while the right column is the estimated perspective map by PACNN. Blue in the heatmaps indicates small perspective values while yellow indicates large values.

GT perspective map generation. *Perspective maps* were widely used in [5, 16, 9, 22, 46, 11]. The GT perspective value at every pixel of the map $P^g = \{p_j^g\}$ is defined as the number of pixels representing one meter at that location in the real scene [46]. The observed object size in the image is thus related to the perspective value. Below we first review the conventional approach to compute the perspective maps in crowded scenes of pedestrians.

Preliminary. Fig. 2 visualizes the perspective geometry of a pinhole camera. Referring to the figure caption, we can solve the similar triangles,

$$\begin{aligned} y_h &= \frac{f(C-H)}{z_1}, \\ y_f &= \frac{fC}{z_1}, \end{aligned} \tag{2}$$

where y_h and y_f are the observed positions of person head and feet on the image plane, respectively. The observed person height h is thus given by,

$$h = y_f - y_h = \frac{fH}{z_1} \tag{3}$$

dividing the two sides of (3) by y_h will give us

$$h = \frac{H}{C-H} y_h. \tag{4}$$

The perspective value p^g is therefore defined as:

$$p^g = \frac{h}{H} = \frac{1}{C-H} y_h. \tag{5}$$

To generate the perspective map for a crowd image, authors in [46] approximate H to be the mean height of adults (1.75m) for every pedestrian. Since C is fixed for each image, p^g becomes a linear function of y_h and remains the same in each row of y_h. To estimate C, they manually labeled the heights h_j of several pedestrians at different positions in each image, such that the perspective value p_j^g at

the sampled position j is given by $p_j^g = \frac{h_j}{1.75}$. They employ a linear regression method afterwards to fit Eqn. (5) and generate the entire GT perspective map.

The perspective maps for datasets WorldExpo'10 [46] and UCSD [5] were generated via the above process. However, for datasets having dense crowds like ShanghaiTech PartA (SHA) [47] and UCF_CC_50 [12], it can not directly apply as the pedestrian bodies are usually not visible in dense crowds. We notice that, similar to the observed pedestrian height, the head size also changes with the perspective distortion. We therefore interpret the sampled perspective value p_j^g by the observed head size, which can be computed following [47] as the average distance from certain head at j to its K-nearest neighbors (K-NN).

The next step is to generate the perspective map based on the sampled values. The conventional linear regression approach [5, 46] relies on several assumptions, e.g. the camera is not in-plane rotated; the ground in the captured scene is flat; the pedestrian height difference is neglected; and most importantly, the sampled perspective values are accurate enough. The first three assumptions are valid for many images in standard crowd counting benchmarks, but there exist special cases such that the camera is slightly rotated; people sit in different tiers of a stadium; and the pedestrian height (head size) varies significantly within a local area. As for the last, using the K-NN distance to approximate the pedestrian head size is surely not perfect; noise exists even in dense crowds as the person distance highly depends on the local crowd density at each position.

Considering the above facts, now we introduce a novel nonlinear way to fit the perspective values, aiming to produce an accurate perspective map that clearly underlines the general head size change from near to far in the image. First, we compute the mean perspective value at each sampled row y_h so as to reduce the outlier influence due to any abrupt density or head size change. We employ a $\tanh$ function to fit these mean values over their row indices y_h:

$$p^g = a \cdot \tanh(b \cdot (y_h + c)), \tag{6}$$

where a, b and c are three parameters to fit in each image. This function produces a perspective map with values decreased from bottom to top and identical in the same row, indicating the vertical person scale change in the image.

The local distance scale has been utilized before to help normalize the detection of traditional method [13]; while in modern CNN-based methods, it is often utilized implicitly in the ground truth density generation [47]. Unlike in [13, 47], the perspective is more than the local distance scale: we mine the reliable perspective information from sampled local scales and fit a nonlinear function over them, which indeed provides additional information about person scale change at every pixel due to the perspective distortion. Moreover, we explicitly encode the perspective map into CNN to guide the density regression at different locations of the image (as below described). The proposed perspective maps are not yet perfect but demonstrated to be helpful (see Sec. 4). On the other hand, if we simply keep the K-NN distance as the final value in the map, we barely get no significant benefit in our experiment.

We generate the GT perspective maps for datasets UCF_CC_50 and ShanghaiTech SHA using our proposed way. While for SHB, the pedestrian bodies are normally visible and the sampled perspective values can be simply obtained by labeling several (less than 10) pedestrian heights; unlike the conventional way, the nonlinear fitting procedure (6) is still applied. We illustrate some examples in Fig. 3 for both SHA and SHB. Notice we also evaluate the linear regression for GT perspective maps in crowd counting, which performs lower than our non-linear way.

3.2. Network architecture

We show the network architecture in Fig. 4: the backbone is adopted from the VGG net [34]; out of Conv4_3, we branch off several data streams to perform the density and perspective regressions, which are described next.

Density map regression. We regress three density maps from the outputs of Conv4_3, Conv5_1_3 and Conv6_1_1 simultaneously. The filters from deeper layers have bigger receptive fields than those from the shallower layers. Normally, a combination of the three density maps is supposed to adapt to varying person size in an image.

We denote by $D^{e_1} = \{d_j^{e_1}\}$, $D^{e_2} = \{d_j^{e_2}\}$ and $D^{e_3} = \{d_j^{e_3}\}$ the three density maps from Conv4_3, Conv5_1_3, and Conv6_1_1, respectively; j signifies the j-th pixel in the map; they are regressed using 1×1 Conv with 1 output. Because of pooling, D^{e_1}, D^{e_2}, and D^{e_3} have different size: D^{e_1} is of $1/8$ resolution of the input, while D^{e_2} and D^{e_3} are of $1/16$ and $1/32$ resolutions, respectively. We downsample the ground truth density map to each corresponding resolution to learn the multi-scale density maps. To combine them, a straightforward way would be averaging their outputs: D^{e_3} is firstly upsampled via a deconvoltuional layer to the same size with D^{e_2}; we denote it by $\mathrm{Up}(D^{e_3})$, $\mathrm{Up}(\cdot)$ is the deconvolutional upsampling; we average $\mathrm{Up}(D^{e_3})$ and D^{e_2} as $(D^{e_2} + \mathrm{Up}(D^{e_3}))/2$; the averaged output is upsampled again and further combined with D^{e_1} to produce the final output D^e:

$$D^e = \frac{D^{e_1} + \mathrm{Up}(\frac{D^{e_2} + \mathrm{Up}(D^{e_3})}{2})}{2} \tag{7}$$

D^e is of $1/8$ resolution of the input, and we need to downsample the corresponding ground truth as well. This combination is a simple, below we introduce our perspective-aware weighting scheme.

Perspective map regression. Perspective maps are firstly regressed in the network. The regression is branched

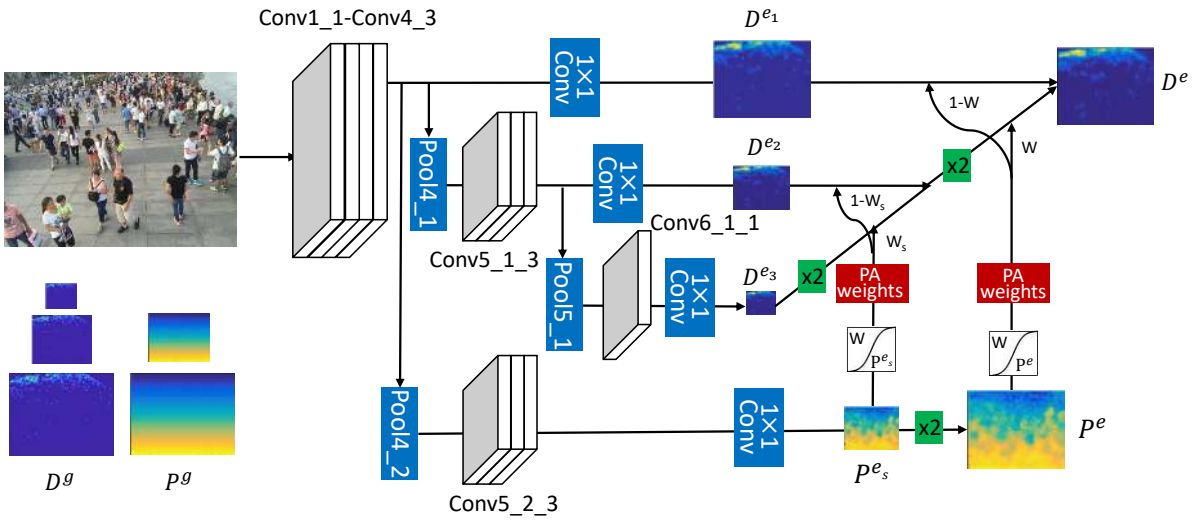

Figure 4: The structure of the proposed perspective-aware convolutional neural network (PACNN). D and P denote the density and perspective map, while e and g stand for estimation and ground truth; green box "$x2$" denotes the deconvolutional layer for upsampling. The backbone is adopted from the VGG net. We regress three density maps D^{e_1}, D^{e_2} and D^{e_3} from Conv4_3, Conv5_1_3 and Conv6_1_1, respectively; two perspective maps P^{e_s} and P^e are produced after Conv5_2_3. We adaptively combine the multi-scale density outputs via two perspective-aware (PA) weighting layers, where the PA weights are learned via the nonlinear transform of P^{e_s} and P^e. We optimize the network over the loss with respect to the ground truth of D^g and P^g in different resolutions. The final density output is D^e.

off from Pool4_2 with three more convolutional layers Conv5_2_1 to Conv5_2_3. We use $P^{e_s} = \{p_j^{e_s}\}$ to denote the regressed perspective map after Conv5_2_3. It is with 1/16 resolution of the input, we further upsample it to 1/8 resolution of the input to obtain the final perspective map $P^e = \{p_j^e\}$. We prepare two perspective maps P^{e_s} and P^e to separately combine the output of D^{e_2} and $\mathrm{Up}(D^{e_3})$, as well as $\mathrm{Up}(D^{e_2} + \mathrm{Up}(D^{e_3}))/2$ and D^{e_1} at different resolutions. Ground truth perspective map is downsampled accordingly to match the estimation size. We present some estimated perspective maps P^e and their corresponding ground truths P^g in Fig. 3.

Perspective-aware weighting. Due to different receptive field size, D^{e_1} is normally good at estimating small heads, D^{e_2} medium heads, while D^{e_3} big heads. We know that the person size in general decreases with an decrease of the perspective value. To make use of the estimated perspective maps P^{e_s} and P^e, we add two perspective-aware (PA) weighting layers in the network (see Fig. 4) to specifically adapt the combination of D^{e_1}, D^{e_2} and D^{e_3} at two levels. The two PA weighting layers work in a similar way to give a density map higher weights on the smaller head area if it is good at detecting smaller heads, and vice versa. We start by formulating the combination between D^{e_2} and $\mathrm{Up}(D^{e_3})$:

$$D^{e_s} = W^s \odot D^{e_2} + (1 - W^s) \odot \mathrm{Up}(D^{e_3}), \quad (8)$$

where $\odot$ denotes the element-wise (Hadamard) product and D^{e_s} the combined output. $W^s = \{w_j^s\}$ is the output of the perspective-aware weighting layer; it is obtained by applying a nonlinear transform $w_j^s = f(p_j^{e_s})$ to the perspective values $p_j^{e_s}$ (nonlinear transform works better than linear transform in our work). This function needs to be differentiable and produce a positive mapping from $p_j^{e_s}$ to w_j^s. We choose the sigmoid function:

$$w_j^s = f(p_j^{e_s}) = \frac{1}{1 + \exp(-\alpha^s * (p_j^{e_s} - \beta^s))}, \quad (9)$$

where α^s and β^s are the two parameters that can be learned via back propagation. $w_j^s \in (0, 1)$, it varies at every pixel of the density map. The backwards function of the PA weighting layer computes partial derivative of the loss function L with respect to α^s and β^s. We will discuss the loss function later. Here we write out the chain rule:

$$\frac{\partial L}{\partial \alpha^s} = \frac{\partial L}{\partial D^{e_s}} \frac{\partial D^{e_s}}{\partial W^s} \frac{\partial W^s}{\partial \alpha^s}$$
$$= \sum_j \frac{\partial L}{\partial d_j^{e_s}} (d_j^{e_2} - \mathrm{Up}(d_j^{e_3}))(p_j^{e_s} - \beta^s) f(p_j^{e_s})(1 - f(p_j^{e_s})); \quad (10)$$

Similarly, we have

$$\frac{\partial L}{\partial \beta^s} = \sum_j \frac{\partial L}{\partial d_j^{e_s}} (d_j^{e_2} - \mathrm{Up}(d_j^{e_3}))(-\alpha^s) f(p_j^{e_s})(1 - f(p_j^{e_s})). \quad (11)$$

The output D^{e_s} can be further upsampled and combined with D^{e_1} using another PA weighting layer:

$$D^e = W \odot D^{e_1} + (1 - W) \odot \mathrm{Up}(D^{e_s}), \quad (12)$$

where $W = \{w_j\}$ is transformed from P^e in a similar way to W^s:

$$w_j = f(p_j^e) = \frac{1}{1 + \exp(-\alpha * (p_j^e - \beta))}, \quad (13)$$

α and β are two parameters similar to α^s and β^s in (9); one can follow (10,11) to write out their backpropagations. Compared to the average operation in (7), which gives the same weights in the combination, the proposed perspective-aware weighting scheme (8) gives different weights on D^{e_1}, D^{e_2} and D^{e_3} at different positions of the image, such that the final output is robust to the perspective distortion.

3.3. Loss function and network training

We regress both the perspective and density maps in a multi-task network. In each specific task O, a typical loss function is the mean squared error (MSE) loss L^{MSE}, which sums up the pixel-wise Euclidean distance between the estimated map and ground truth map. The MSE loss does not consider the local correlation in the map, likewise in [4], we adopt the DSSIM loss to measure the local pattern consistency between the estimated map and ground truth map. The DSSIM loss L^{DSSIM} is derived from the structural similarity (SSIM) [43]. The whole loss for task O is thereby,

$$\begin{aligned} L_{\mathrm{O}}(\Theta) &= L^{\mathrm{MSE}} + \lambda L^{\mathrm{DSSIM}} \\ &= \frac{1}{2N}\sum_{i=1}^{N} \|E(X_i;\Theta) - G_i\|_2^2 \\ &+ \lambda \frac{1}{N}\sum_{i=1}^{N}(1 - \frac{1}{M}\sum_{j}\mathrm{SSIM}_i(j)) \\ \mathrm{SSIM}_i &= \frac{(2\mu_{E_i}\mu_{G_i} + C_1)}{\mu_{E_i}^2 + \mu_{G_i}^2 + C_1} \cdot \frac{(2\sigma_{E_iG_i} + C_2)}{\sigma_{E_i}^2 + \sigma_{G_i}^2 + C_2} \end{aligned} \tag{14}$$

where Θ is a set of learnable parameters in the proposed network; X_i is the input image, N is the number of training images and M is the number of pixels in the maps; λ is the weight to balance L^{MSE} and L^{DSSIM}. We denote by E and G the respective estimated map and ground truth map for task O. Means (μ_{E_i}, μ_{G_i}) and standard deviations (σ_{E_i}, σ_{G_i}, $\sigma_{E_iG_i}$) in SSIM_i are computed with a Gaussian filter with standard deviation 1 within a 5×5 region at each position j. We omit the dependence of means and standard deviations on pixel j in the equation.

For the perspective regression task P, we obtain its loss L_{P} from (14) by substituting P^e and P^g into E and G, respectively; while for the density regression task D, we obtain its loss L_{D} by replacing E and G with D^e and D^g correspondingly. We offer our overall loss function as

$$L = L_{\mathrm{P}} + L_{\mathrm{D}} + \kappa L_{\mathrm{P}^s} + \lambda_1 L_{\mathrm{D}^1} + \lambda_2 L_{\mathrm{D}^2} + \lambda_3 L_{\mathrm{D}^3}. \tag{15}$$

As mentioned in Sec. 3.2, L_{P^s} is a subloss for P^{e_s} while L_{D^1}, L_{D^2} and L_{D^3} are the three sub-losses for D^{e_1}, D^{e_2} and D^{e_3}. We empirically give small loss weights for these sublosses. We notice that the ground truth perspective and density maps are pre-processed to have the same scale in practice. The training is optimized with Stochastic Gradient Descent (SGD) in two phases. **Phase 1**: we optimize the density regression using the architecture in Fig 5; **Phase 2**: we finetune the model by adding the perspective-aware weighting layers to jointly optimize the perspective and density regressions.

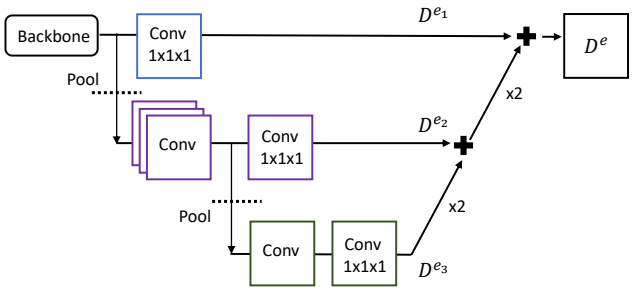

Figure 5: Network architecture without using perspective (denoted as PACNN w/o P). Referring to (7), multi-scale density outputs are adapted to the same resolution and averaged to produce the final prediction.

4. Experiments

4.1. Datasets

ShanghaiTech [47]. It consists of 1,198 annotated images with a total of 330,165 people with head center annotations. This dataset is split into two parts SHA and SHB. The crowd images are sparser in SHB compared to SHA: the average crowd counts are 123.6 and 501.4, respectively. Following [47], we use 300 images for training and 182 images for testing in SHA; 400 images for training and 316 images for testing in SHB.

WorldExpo'10 [46]. It includes 3,980 frames, which are taken from the Shanghai 2010 WorldExpo. 3,380 frames are used as training while the rest are taken as test. The test set includes five different scenes and 120 frames in each one. Regions of interest (ROI) are provided in each scene so that crowd counting is only conducted in the ROI in each frame. The crowds in this dataset are relatively sparse with an average pedestrian number of 50.2 per image.

UCF_CC_50 [12]. It has 50 images with 63,974 head annotations in total. The head counts range between 94 and 4,543 per image. The small dataset size and large count variance make it a very challenging dataset. Following [12], we perform 5-fold cross validations to report the average test performance.

UCSD [5]. This dataset contains 2000 frames chosen from one surveillance camera in the UCSD campus. The frame size is 158 $\times$ 238 and it is recorded at 10 fps. There are only about 25 persons on average in each frame. It provides the ROI for each video frame. Following [5], we use frames from 601 to 1400 as training data, and the remaining 1200 frames as test data.

4.2. Implementation details and evaluation protocol

Ground truth annotations for each head center are publicly available in the standard benchmarks. For Wolrd-Expo'10 and UCSD, the ground truth perspective maps are provided. For ShanghaiTech and UCF, ground truth perspective maps are generated as described in Sec. 3.1[1]. Given a training set, we augment it by randomly cropping 9 patches from each image. Each patch is $1/4$ size of the original image. All patches are used to train our PACNN. The backbone is adopted from VGG-16 [34], pretrained on ILSVRC classification data. We set the batch size as 1, learning rate 1e-6 and momentum 0.9. We train 100 epochs in Phase 1 while 150 epochs in Phase 2 (Sec. 3.3). Network inference is on the entire image.

We evaluate the performance via the mean absolute error (MAE) and mean squared error (MSE) as commonly used in previous works [46, 47, 31, 25, 41]: Small MAE and MSE values indicate good performance.

4.3. Results on ShanghaiTech

Ablation study. We conduct an ablation study to justify the utilization of multi-scale and perspective-aware weighting schemes in PACNN. Results are shown in Table 1.

Referring to Sec. 3.2, D^{e_1}, D^{e_1} and D^{e_3} should fire more on small, medium and big heads, respectively. Having a look at Table 1, the MAE for D^{e_1} D^{e_2} and D^{e_3} on SHA are 81.8, 86.3 and 93.1, respectively; on SHB they are 16.0, 14.5 and 18.2, respectively. Crowds in SHA are much denser than in SHB, persons are mostly very small in SHA and medium/medium-small in SHB. It reflects in Table 1 that D^{e_1} in general performs better on SHA while D^{e_2} performs better on SHB.

To justify the PA weighting scheme, we compare PACNN with the average weighting scheme (see Fig. 5) in Table 1. Directly averaging over pixels of D^{e_1} and upsampled D^{e_2} and D^{e_3} (PACNN w/o P) produces a marginal improvement of MAE and MSE on SHA and SHB. For instance, the MAE is decreased to 76.5 compared to 81.8 of D^{e_1} on SHA; 12.9 compared to 14.5 of D^{e_2} on SHB. In contrast, using PA weights to adaptively combine D^{e_1}, D^{e_1} and D^{e_3} significantly decreases the MAE and MSE on SHA and SHB: they are 66.3 and 106.4 on SHA; 8.9 and 13.5 on SHB, respectively.

Comparison to state-of-the-art. We compare PACNN with the state-of-the-art [36, 32, 20, 17, 28, 4] in Table 1. PACNN produces the lowest MAE 66.3 on SHA and lowest MSE 13.5 on SHB, the second lowest MSE 106.4 on SHA and MAE 8.9 on SHB compared to the previous best results [4, 32]. We notice that many previous methods employ the patch-based inference [36, 32, 4], where model inference is usually conducted with a sliding window strategy. We illustrate the inference type for each method in Table 1. Patch-based inference can be very time-consuming factoring the additional cost to crop and resize patches from images and merge their results. On the other hand, PACNN employs an image-based inference and can be very fast; for instance, in the same Caffe [14] framework with an Nvidia GTX Titan X GPU, the inference time of our PACNN for an 1024*768 input is only 230ms while those with patch-based inference can be much (e.g. 5x) slower in our experiment. If we compare our result to previous best result with the image-based inference (e.g. [17]), ours is clearly better. We can further combine our method with [17] by adopting its trained backbone, we achieve the lowest MAE and MSE: 62.4 and 102.0 on SHA, 7.6 and 11.8 on SHB. This demonstrates the robustness and efficiency of our method in a real application. Fig. 6 shows some examples.

ShanghaiTech	Inference	SHA		SHB	
		MAE	MSE	MAE	MSE
D^{e_1}	image	81.8	131.1	16.0	21.9
D^{e_2}	image	86.3	138.6	14.5	18.7
D^{e_3}	image	93.1	156.4	18.2	25.1
PACNN w/o P	image	76.5	123.3	12.9	17.2
PACNN	image	66.3	106.4	8.9	13.5
PACNN + [17]	image	**62.4**	**102.0**	**7.6**	**11.8**
Cao et al. [4]	patch	**67.0**	104.5	**8.4**	**13.6**
Ranjan et al. [28]	image*	68.5	116.2	10.7	16.0
Li et al. [17]	image	68.2	115.0	10.6	16.0
Liu et al. [20]	-	73.6	112.0	13.7	21.4
Shen et al. [32]	patch	75.7	**102.7**	17.2	27.4
Sindagi et al. [36]	patch	73.6	106.4	20.1	30.1

Table 1: Ablation study of PACNN and its comparison with state-of-the-art on ShanghaiTech dataset. D^{e_1}, D^{e_2} and D^{e_3} denote the density map regressed from Conv4_3, Conv5_1_3 and Conv6_1_1 in Fig. 4, respectively. "Inference" signifies whether it is patch-based or image-based. "-" means it is not mentioned in the paper. "image*" denotes that a two-stage inference in [28]. PACNN w/o P denotes our network without using perspective maps (see Fig. 5).

4.4. Results on UCF_CC_50

We compare our method with other state-of-the-art on UCF_CC_50 [36, 20, 17, 28, 4] in Table 2. Our method PACNN achieves the MAE 267.9 and MSE 357.8; while the best MAE is 258.4 from [4] and MSE 320.9 from [36]. We also present the result of PACNN + [17], which produces the lowest MAE and MSE: 241.7 and 320.7. We notice the backbone model of [17] that we use to combine with PACNN is trained by ourselves. Our reproduced model produces slightly lower MAE and MSE (262.5 and 392.7) than the results in [17].

[1]Ground truth perspective maps for ShanghaiTech can be downloaded from here: https://drive.google.com/open?id=117MLmXj24-vg4Fz0MZcm9jJISvZ46apK

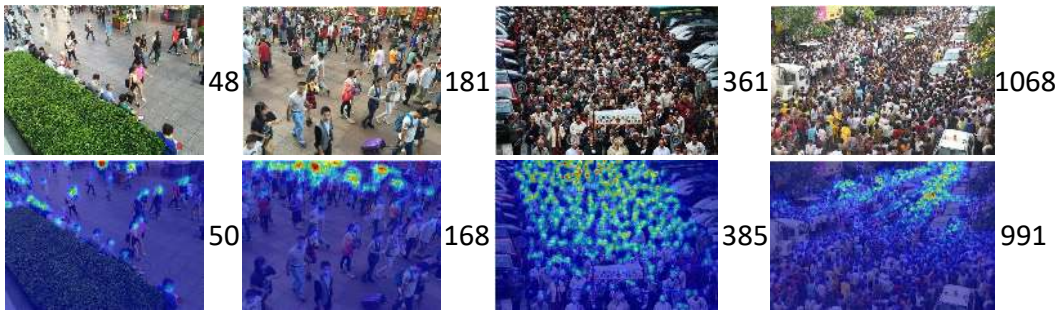

Figure 6: Results on ShanghaiTech dataset. We present four test images and their estimated density maps below. The ground truth and estimated crowd counts are to the right of the real images and the corresponding density maps, respectively.

UCF_CC_50	MAE	MSE
Sindgai et al. [36]	295.8	320.9
Liu et al. [20]	279.6	388.5
Li et al. [17]	266.1	397.5
Ranjan et al. [28]	260.9	365.5
Cao et al. [4]	258.4	344.9
PACNN	267.9	357.8
PACNN + [17]	**241.7**	**320.7**

Table 2: Comparison of PACNN with other state-of-the-art on UCF_CC_50 dataset.

WorldExpo'10	S1	S2	S3	S4	S5	Avg.
Sindagi et al. [36]	2.9	14.7	10.5	10.4	5.8	8.9
Xiong et al. [45]	6.8	14.5	14.9	13.5	3.1	10.6
Li et al. [17]	2.9	**11.5**	**8.6**	16.6	3.4	8.6
Liu et al. [20]	**2.0**	13.1	8.9	17.4	4.8	9.2
Ranjan et al. [28]	17.0	12.3	9.2	**8.1**	4.7	10.3
Cao et al. [4]	2.6	13.2	9.0	13.3	**3.0**	8.2
PACNN	2.3	12.5	9.1	11.2	3.8	**7.8**

Table 3: Comparison of PACNN with other state-of-the-art on WorldExpo'10 dataset. MAE is reported for each test scene and averaged in the end.

4.5. Results on WorldExpo'10

Referring to [46], training and test are both conducted within the ROI provided for each scene of WorldExpo'10. MAE is reported for each test scene and averaged to evaluate the overall performance. We compare our PACNN with other state-of-the-art [36, 45, 19, 17, 28, 4] in Table 3. It can be seen that although PACNN does not outperform the state-of-the-art in each specific scene, it produces the lowest mean MAE 7.8 over the five scenes. Perspective information is in general helpful for crowd counting in various scenarios.

4.6. Results on UCSD

The crowds in this dataset is not evenly distributed and the person scale changes drastically due to the perspective distortion. Perspective maps were originally proposed in this dataset to weight each image location in the crowd segment according to its approximate size in the real scene. We evaluate our PACNN in Table 4: comparing to the state-of-the-art [47, 25, 11, 31, 17, 4], PACNN significantly decreases the MAE and MSE to the lowest: 0.89 and 1.18, which demonstrates the effectiveness of our perspective-aware framework. Besides, the crowds in this dataset is in general sparser than in other datasets, which shows the generalizability of our method over varying crowd densities.

UCSD	MAE	MSE
Zhang et al. [47]	1.60	3.31
Onoro et al. [25]	1.51	-
Sam et al. [31]	1.62	2.10
Huang et al. [11]	1.00	1.40
Li et al. [17]	1.16	1.47
Cao et al. [4]	1.02	1.29
PACNN	**0.89**	**1.18**

Table 4: Comparison of PACNN with other state-of-the-art on UCSD dataset.

5. Conclusion

In this paper we propose a perspective-aware convolutional neural network to automatically estimate the crowd counts in images. A novel way of generating GT perspective maps is introduced for PACNN training, such that at the test stage it predicts both the perspective maps and density maps. The perspective maps are encoded as two perspective-aware weighting layers to adaptively combine the multi-scale density outputs. The combined density map is demonstrated to be robust to the perspective distortion in crowd images. Extensive experiments on standard crowd counting benchmarks show the efficiency and effectiveness of the proposed method over the state-of-the-art.

Acknowledgments. This work was supported by NSFC 61828602 and 61733013. Zhaohui Yang and Chao Xu were supported by NSFC 61876007 and 61872012. We thank Dr. Yannis Avrithis for the discussion on perspective geometry and Dr. Holger Caesar for proofreading.

References

[1] Carlos Arteta, Victor Lempitsky, and Andrew Zisserman. Counting in the wild. In *ECCV*, 2016. 3

[2] Lokesh Boominathan, Srinivas SS Kruthiventi, and R Venkatesh Babu. Crowdnet: a deep convolutional network for dense crowd counting. In *ACM MM*, 2016. 2

[3] Gabriel J Brostow and Roberto Cipolla. Unsupervised bayesian detection of independent motion in crowds. In *CVPR*, 2006. 1, 2

[4] Xinkun Cao, Zhipeng Wang, Yanyun Zhao, and Fei Su. Scale aggregation network for accurate and efficient crowd counting. In *ECCV*, 2018. 2, 3, 6, 7, 8

[5] Antoni B Chan, Zhang-Sheng John Liang, and Nuno Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *CVPR*, 2008. 1, 2, 3, 4, 6

[6] Antoni B Chan and Nuno Vasconcelos. Counting people with low-level features and bayesian regression. *IEEE Transactions on Image Processing*, 21(4):2160–2177, 2012. 1, 2

[7] Ke Chen, Chen Change Loy, Shaogang Gong, and Tony Xiang. Feature mining for localised crowd counting. In *BMVC*, 2012. 1, 2

[8] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2

[9] Luca Fiaschi, Ullrich Köthe, Rahul Nair, and Fred A Hamprecht. Learning to count with regression forest and structured labels. In *ICPR*, pages 2685–2688, 2012. 1, 3

[10] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):930–943, 2003. 1

[11] Siyu Huang, Xi Li, Zhongfei Zhang, Fei Wu, Shenghua Gao, Rongrong Ji, and Junwei Han. Body structure aware deep crowd counting. *IEEE Transactions on Image Processing*, 27(3):1049–1059, 2018. 1, 3, 8

[12] Haroon Idrees, Imran Saleemi, Cody Seibert, and Mubarak Shah. Multi-source multi-scale counting in extremely dense crowd images. In *CVPR*, 2013. 1, 2, 4, 6

[13] Haroon Idrees, Khurram Soomro, and Mubarak Shah. Detecting humans in dense crowds using locally-consistent scale prior and global occlusion reasoning. *TPAMI*, 37(10):1986–1998, 2015. 2, 3, 4

[14] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014. 7

[15] Dan Kong, Douglas Gray, and Hai Tao. Counting pedestrians in crowds using viewpoint invariant training. In *BMVC*, 2005. 1, 2

[16] Victor Lempitsky and Andrew Zisserman. Learning to count objects in images. In *NIPS*, 2010. 1, 2, 3

[17] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *CVPR*, 2018. 2, 7, 8

[18] Sheng-Fuu Lin, Jaw-Yeh Chen, and Hung-Xin Chao. Estimation of number of people in crowded scenes using perspective transformation. *TSMC-A*, 31(6):645–654, 2001. 2

[19] Jiang Liu, Chenqiang Gao, Deyu Meng, and Alexander G. Hauptmann. Decidenet: Counting varying density crowds through attention guided detection and density estimation. In *CVPR*, 2018. 2, 8

[20] Xialei Liu, Joost Weijer, and Andrew D Bagdanov. Leveraging unlabeled data for crowd counting by learning to rank. In *CVPR*, 2018. 2, 7, 8

[21] Yuting Liu, Miaojing Shi, Qijun Zhao, and Xiaofang Wang. Point in, box out: Beyond counting persons in crowds. In *CVPR*, 2019. 1, 2

[22] Chen Change Loy, Ke Chen, Shaogang Gong, and Tao Xiang. Crowd counting and profiling: Methodology and evaluation. In *Modeling, Simulation and Visual Analysis of Crowds*, pages 347–382. Springer, 2013. 1, 2, 3

[23] Zhang Lu, Miaojing Shi, and Qiaobo Chen. Crowd counting via scale-adaptive convolutional neural network. In *WACV*, 2018. 2

[24] AN Marana, L da F Costa, RA Lotufo, and SA Velastin. On the efficacy of texture analysis for crowd monitoring. In *SIBGRAPI*, 1998. 2

[25] Daniel Onoro-Rubio and Roberto J López-Sastre. Towards perspective-free object counting with deep learning. In *ECCV*, 2016. 1, 2, 7, 8

[26] Nikos Paragios and Visvanathan Ramesh. A mrf-based approach for real-time subway monitoring. In *CVPR*, 2001. 2

[27] Vincent Rabaud and Serge Belongie. Counting crowded moving objects. In *CVPR*, 2006. 1, 2

[28] Viresh Ranjan, Hieu Le, and Minh Hoai. Iterative crowd counting. *ECCV*, 2018. 2, 7, 8

[29] Carlo S Regazzoni and Alessandra Tesei. Distributed data fusion for real-time crowding estimation. *Signal Processing*, 53(1):47–63, 1996. 2

[30] David Ryan, Simon Denman, Clinton Fookes, and Sridha Sridharan. Crowd counting using multiple local features. In *DICTA*, 2009. 1, 2

[31] Deepak Babu Sam, Shiv Surya, and R Venkatesh Babu. Switching convolutional neural network for crowd counting. In *CVPR*, 2017. 1, 2, 3, 7, 8

[32] Zan Shen, Yi Xu, Bingbing Ni, Minsi Wang, Jianguo Hu, and Xiaokang Yang. Crowd counting via adversarial cross-scale consistency pursuit. In *CVPR*, 2018. 2, 7

[33] Zenglin Shi, Le Zhang, Yun Liu, Xiaofeng Cao, Yangdong Ye, Ming-Ming Cheng, and Guoyan Zheng. Crowd counting with deep negative correlation learning. In *CVPR*, 2018. 2

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 4, 7

[35] Vishwanath A Sindagi and Vishal M Patel. Cnn-based cascaded multi-task learning of high-level prior and density estimation for crowd counting. In *AVSS*, 2017. 2

[36] Vishwanath A Sindagi and Vishal M Patel. Generating high-quality crowd density maps using contextual pyramid cnns. In *ICCV*, 2017. 2, 3, 7, 8

[37] Russell Stewart, Mykhaylo Andriluka, and Andrew Y Ng. End-to-end people detection in crowded scenes. In *CVPR*, 2016. 2

[38] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001. 2

[39] Paul Viola, Michael J Jones, and Daniel Snow. Detecting pedestrians using patterns of motion and appearance. *IJCV*, 63(2):153–161, 2003. 1, 2

[40] Elad Walach and Lior Wolf. Learning to count with cnn boosting. In *ECCV*, 2016. 2

[41] Chuan Wang, Hua Zhang, Liang Yang, Si Liu, and Xiaochun Cao. Deep people counting in extremely dense crowds. In *ACM MM*, 2015. 7

[42] Meng Wang and Xiaogang Wang. Automatic adaptation of a generic pedestrian detector to a specific traffic scene. In *CVPR*, 2011. 2

[43] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 6

[44] Bo Wu and Ramakant Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *ICCV*, 2005. 1, 2

[45] Feng Xiong, Xingjian Shi, and Dit-Yan Yeung. Spatiotemporal modeling for crowd counting in videos. In *ICCV*, 2017. 2, 8

[46] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. In *CVPR*, 2015. 1, 2, 3, 4, 6, 7, 8

[47] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *CVPR*, 2016. 1, 2, 3, 4, 6, 7, 8

[48] Zhuoyi Zhao, Hongsheng Li, Rui Zhao, and Xiaogang Wang. Crossing-line crowd counting with two-phase deep neural networks. In *ECCV*, 2016. 2