Crossing-Line Crowd Counting with Two-Phase Deep Neural Networks

Zhuoyi Zhao[1], Hongsheng Li[1(✉)], Rui Zhao[2,3], and Xiaogang Wang[1(✉)]

[1] Department of Electronic Engineering,
The Chinese University of Hong Kong, Shatin, Hong Kong
{zyzhao,hsli,xgwang}@ee.cuhk.edu.hk
[2] SenseNets Technology Limited, Shenzhen, China
zhaorui@sensenets.com
[3] SenseTime Group Limited, Shatin, Hong Kong

Abstract. In this paper, we propose a deep Convolutional Neural Network (CNN) for counting the number of people across a line-of-interest (LOI) in surveillance videos. It is a challenging problem and has many potential applications. Observing the limitations of temporal slices used by state-of-the-art LOI crowd counting methods, our proposed CNN directly estimates the crowd counts with pairs of video frames as inputs and is trained with pixel-level supervision maps. Such rich supervision information helps our CNN learn more discriminative feature representations. A two-phase training scheme is adopted, which decomposes the original counting problem into two easier sub-problems, estimating crowd density map and estimating crowd velocity map. Learning to solve the sub-problems provides a good initial point for our CNN model, which is then fine-tuned to solve the original counting problem. A new dataset with pedestrian trajectory annotations is introduced for evaluating LOI crowd counting methods and has more annotations than any existing one. Our extensive experiments show that our proposed method is robust to variations of crowd density, crowd velocity, and directions of the LOI, and outperforms state-of-the-art LOI counting methods.

Keywords: Intelligent surveillance · Crowd counting · Deep learning

1 Introduction

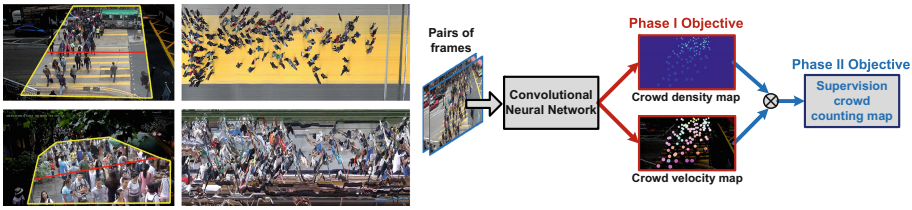
Automatically counting crowds in surveillance scenes with large population density, such as train stations, shopping malls, and tourist attractions, has escalating demands in intelligent surveillance industry. It has drawn increasing attention from the computer vision community in recent years and can provide vital information for the management departments of public spaces to make crucial decisions. Nearly all major stampedes resulted from the failure of controlling crowd density and traffic flow in public spaces. Such problems could be effectively mitigated by intelligent surveillance systems with crowd counting capabilities. Whenever the crowd number of a public space exceeds an alarming threshold,

B. Leibe et al. (Eds.): ECCV 2016, Part VIII, LNCS 9912, pp. 712–726, 2016.
DOI: 10.1007/978-3-319-46484-8_43

the management departments can be automatically alerted to control crowd density by either opening new exits, closing entrances, or sending extra staffs to guide the crowd traffic flows. However, the crowd counting problem is challenging, especially for scenes with high crowd density and low viewing angles. The pedestrians might heavily occlude each other, which prevent pedestrian tracking methods from being used to solve this problem.

The existing crowd counting algorithms either count the number of people in a region-of-interest (ROI) or count the number of people crossing a line-of-interest (LOI) (see Fig. 1(a)). Although most algorithms focus on the former ROI counting problem, solving the latter LOI counting problem has more practical uses. For instance, in order to count the number of people in a public space by ROI counting algorithms, all locations of the space should be monitored by cameras, which might not be practical for places like large squares and metro stations. In the contrary, the LOI counting systems only need to monitor the entrances and exits of a space. By counting the number of people across the lines of interest, the total number of people within the space can be easily inferred.

State-of-the-art methods for solving the crossing-line crowd counting problem require generating 2D temporal slices by temporally concatenating video frame lines at the LOI (see Fig. 1(b) for examples). The number of people across the line is then estimated based on the temporal slices. When the scene is not crowded and pedestrians walks in normal speed (Fig. 1(b, row 1)), people can be well recognized in the temporal slices. However, we observe that temporal slices are not robust to scenes with high crowd density, slow walking speed, and low camera viewing angles. In Fig. 1(b, row2), the temporal slice shows excess jitters and people in it are no longer recognizable.



(a) LOI (red) (b) Temporal slices (c) Our two-phase training strategy

Fig. 1. (a) The problem of counting people crossing a line-of-interest (red) in both directions. (b) Temporal slices generated from (a), which are used by state-of-the-art LOI counting methods. (Row 1) When the scene is not crowded and pedestrians' walking speed is not too slow, people could be distinguished from the temporal slices. (Row 2) The temporal slices are heavily degenerated when the scenes are very crowded. (c) Our proposed Convolutional Neural Network (CNN) is trained to solve the crossing-line people counting problem with rich pixel-level supervision maps in two phases. In the first phase, the CNN learns to predict the crowd density map and crowd velocity map. In the second phase, the CNN learns to predict crowd counting map in an end-to-end manner. (Color figure online)

In this paper, we propose a deep Convolutional Neural Network (CNN) for counting the number of people across the line of interest by directly using video frames as input. Deep learning models usually learn better if rich supervision information is provided. We therefore generate rich pixel-level supervision from annotated pedestrian trajectories for training our CNN. Three types of supervision maps are generated, crowd density maps, crowd velocity maps, and crossing-line crowd counting maps. Based on the new learning objectives, a two-phase training scheme is designed (Fig. 1(c)). Our network is first trained to simultaneously estimate the crowd density map and crowd velocity map, where the two tasks share the same bottom neural layers. As shown by our experiments, sharing the same bottom neural layers helps learn more effective feature representations for each of the tasks. The estimated crowd density and crowd velocity maps are then elementwisely multiplied to generate the crowd counting maps and trained with only the supervision crowd counting maps in an end-to-end manner.

In comparison, existing methods utilize only the number of crossing-line people as ground truth for training. Much spatial information is ignored and the mapping from features to final counts is treated as a black box. In contrast, the three types of our supervision are all of pixel-level and help better train the deep model. Each of our two learning phases has specific physical meanings: the first step is to learn pixel-level crowd density and crowd velocity, and the second step is to learn pixel-level crossing-line crowd counts. Such a strategy can be viewed as decomposing the original crowd counting problem to two easier sub-problems and optimizing our model to solve them first. This strategy avoids directly recovering complex and highly non-linear relations between video frames and final per-pixel crowd counts. In addition, by learning to estimate pixel-level crossing-line counting maps for different scenes, our proposed model is robust to the variations of scene appearances.

The contribution of this paper is three-fold. (1) Observing the limitations of temporal slices used by state-of-the-art methods, we propose training a CNN model that directly estimates crossing-line crowd counts from video frames. Since CNN models generally learns better with rich supervision signals, pixel-level supervision maps are designed for training. (2) A two-phase optimization scheme is proposed to first train the CNN model on the two easier sub-problems of crowd density estimation and crowd velocity estimation. It is then optimized to solve crossing-line crowd counting task based on the results of the sub-problems in an end-to-end manner. (3) We contribute a large-scale dataset for evaluating crossing-line crowd counting algorithms, which includes 5 different scenes, 3,100 annotated frames and 5,900 annotated pedestrians. Compared with existing datasets, our dataset has more scenes and provides trajectory annotations for each pedestrian instead of crowd counts of fixed lines. Counting ground truth at different LOIs could be easily obtained for our dataset.

2 Related Work

The learning-based crowd counting algorithms can be generally categorized into two types: the first type of methods focus on counting the number of people in

a region-of-interest (ROI), and the second type of methods learn to count the number of people across a line-of-interest (LOI).

ROI Counting. Most existing ROI counting methods extract features from the ROI and learn a regression function to predict the number of people within it. Chan *et al.* [3] identified pedestrian foreground by motion segmentation. A Gaussian process regression function is trained to predict the counts based on the shape, edge, and texture features of the pedestrian foreground. In [4], the same features as those in [3] were used, but a Bayesian Poisson regression function is trained to predict the crowd counts. In [13], linear regression functions are learned to predict the number of people in small blobs with low-level features, and the overall number is obtained by summing up the numbers of all small blobs.

There were also methods that learn to estimate pixel-level density maps for a ROI instead of just predicting a single count number. The counting number could be obtained by integrating over the estimated density values in the ROI. Lempitsky and Zisserman [8] proposed to generate ground truth density maps with Gaussian functions and learn linear regression functions for estimating per-pixel density with a specially designed loss function. Zhang *et al.* [18] proposed a CNN for cross-scene crowd counting, which is trained alternatively to learn the crowd density map and the crowd counting numbers.

LOI counting. To the best of our knowledge, all state-of-the-art methods [2,5,10] only use the numbers of crossing-line pedestrians as supervision. Where and how fast a pedestrian has crossed the LOI are ignored. In addition, all the methods require to create temporal slices before counting, which are created by temporally concatenating the intensity values on the LOI. The limitations of such temporal slices are illustrated in Fig. 1(b). When creating temporal slices, the flow mosaicking method [5] assigns different thickness at different locations of the LOI based on the traffic flow velocity. Pedestrians with different velocities could be "normalized" to generate images of similar appearance. A quadratic regression function is then trained to predict the final crossing-line counts. However, the major limitation of flow mosaicking is that the flow velocity at LOI might not be reliably obtained for complex scenes. Ma and Chan [10] proposed an integer programming method for estimating crossing-line crowd counts. Local HOG features are extracted from moving foreground regions of the temporal slices to train a Gaussian process regression function. In [2], deep learning models are trained for the LOI counting problem. Given the image temporal slices and optical flow temporal slices, three CNNs are trained to estimate the number of people in a temporal slice, the type of a temporal slice, and the ratio of the numbers of people crossing the LOI in the two directions, respectively. By combining the results of the three CNNs, the number of pedestrians across the LOI in both directions can be obtained.

Multi-person tracking. The crowd counting problem might also be solved by multi-person tracking methods, such as [1,11,12], if the scene is not crowded and people do not heavily occlude each other. The number of people across a LOI

in both directions can be easily obtained by counting the number of pedestrian trajectories that crossed the LOI. However, such pedestrian tracking methods do not work well when the scene is crowded. Pedestrian-to-pedestrian occlusions severely affected the tracking performance and led to poor counting results.

Deep learning. Deep convolutional neural networks have achieved great success on image classification [14] and object detection [15]. For surveillance applications, various deep learning methods were proposed for tasks including person re-identification [9], pedestrian detection [16] and object tracking [17]. The only deep learning method for LOI crowd counting [2] is based on temporal slices and is therefore unreliable when the scene is too crowded or the camera has a too low viewing angle. The FlowNet [6] was proposed to estimate optical flow from pairs of images, which is used as the base model for our proposed CNN model.

3 Method

3.1 Pixel-Level Supervision Maps

Deep Convolutional Neural Networks (CNN) learn more discriminative features when rich supervision information is provided. For instance, pre-training a CNN for 1,000-class image classification on the ImageNet dataset is able to boost the performance of 200-class object detection task by a large margin [7].

Unlike exiting LOI counting methods that utilize single crossing-line counting numbers on an LOI as training supervision signals, the learning goal for our CNN is designed as pixel-level crossing-line crowd counting map C_t at each time t, where each location of the map records a two-dimensional vector representing how many people pass this location along x and y directions respectively between t and $t+1$ (see Fig. 2(b) for illustration). Note that the values of each entry are generally smaller than 1, which denotes that a fraction of a person have passed this location at time t. When an LOI is provided, the crossing-line crowd counts at each time t could be obtained by first transforming each location's x- and y-directional crowd counts to the normal direction of the LOI and then integrating the normalized values on the LOI as the final count.

However, raw crowd counting map directly obtained from pedestrian annotations (Fig. 2(a)) would not be suitable for training the CNN because of its high annotation sparsity. We therefore model the crossing-line crowd counting map as elementwise multiplication of a crowd density map D_t and a crowd velocity map V_t, *i.e.* $C_t = D_t \otimes V_t$. This decomposition assumes that pedestrians' velocity and density on the LOI remain constant between time t and $t+1$. Since our proposed algorithm works on videos of frame rate greater than 25 fps, the time interval is small enough for generating accurate crowd counting map with such crowd density and crowd velocity maps.

Given dot pedestrian annotations $\mathbf{P}_t = \{P_t^1, \cdots, P_t^n\}$ at time t (Fig. 2(a)), the crowd density map D_t (Fig. 2(c)) can be generated by applying a kernel density estimation function based on the annotated pedestrian locations as

$$D_t(p) = \sum_{P \in \mathbf{P}_t} \mathcal{N}(p; P, \sigma_P^2), \tag{1}$$

(a) Dot annotations (b) Counting map (c) Density map (d) Velocity map

Fig. 2. (a) Dot annotations on pedestrians. (b) Illustration of the crossing-line crowd counting map, where each entry stores how many people pass this location along x and y directions. (c) The supervision crowd density map generated based on (a). Hotter color denotes higher crowd density. (d) The supervision crowd velocity map generated based on (a). (Color figure online)

where $D_t(p)$ denotes the density value at location p, $\mathcal{N}(p; P, \sigma_P^2)$ denotes a normalized 2D Gaussian kernel evaluated at p, with a mean value at location P and a bandwidth σ_P. For each scene, a perspective map could be obtained following [3]. The bandwidth σ_P is perspectively normalized based on the dot location P to represent objects at different distances to the camera. The sum of such density values in any region would be the same as counting the crowd within it. The generated crowd density map is similar to those in [8,18].

To generate the crowd velocity maps V_t (Fig. 2(d)) from dot pedestrian annotations $\mathbf{P}_t$ and $\mathbf{P}_{t-1}$, we first calculate the x- and y-displacements between corresponding pedestrian annotations $v_t^{P^i} = P_t^i - P_{t-1}^i$. The crowd velocity map V_t is then created as

$$V_t(p) = \sum_{P \in \mathbf{P}_t} K(p; P, r_P) \cdot v_t^P, \tag{2}$$

where $V_t(p)$ stores the crowd velocity at location p along x and y directions. $K(p; P, r_P)$ is a disk-shape masking function with a perspectively normalized radius threshold r_P that assigns each dot annotation's displacement values v_t^P to the small neighborhood around it.

$$K(p; P, r_P) = \mathbf{1}\left(p \leq \|P - r_P\|^2\right), \tag{3}$$

where $\mathbf{1}(\cdot)$ represents the indicator function. Note that our crowd velocity map is different from the general optical flow maps and only has non-zero flow values at pedestrian locations.

To obtain a crowd counting map for training our CNN, one could first generate the crowd density map and crowd velocity map as (2) and (3) based on the ground truth dot annotations. The crowd counting map can then be obtained as the elementwise multiplication of the density and velocity maps. In Sect. 4.3, we show that the supervision counting map obtained in this way is accurate. Therefore, we choose to use those pixel-level maps as supervision signals for training our CNN model.

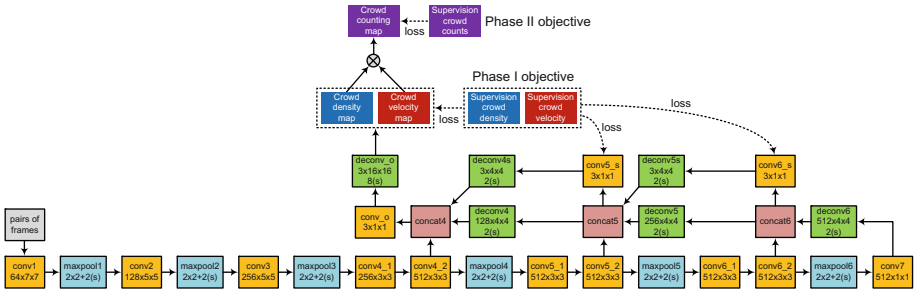


Fig. 3. Illustration of our proposed CNN model for crossing-line crowd counting. "conv", "maxpool", "deconv", and "concat" represent convolution, max pooling, deconvolutional, and channel-wise concatenation layers, respectively. "$c \times k \times k + m(s)$" denotes that a layer has c kernels of size $k \times k$ with a stride m. The output of each convolution layer is activated by a Rectified Linear Unit (ReLU) function. In the first phase, the learning supervisions for our CNN model are the crowd density map and crowd velocity map introduced in Sect. 3.1. In the second phase, the learning supervision is the final crowd counting map. The dashed arrows represent training supervision signals. (Color figure online)

3.2 Deep Convolutional Neural Network for LOI Crowd Counting

CNN models have shown great capabilities on learning discriminative feature representations for various computer vision tasks. An overview of our LOI crowd counting CNN model is shown in Fig. 3, which consists of convolutional layers, max pooling layers and deconvolutional layers. The network structure is similar to that of the FlowNetSimple network in [6] but with max pooling layers. The input of the CNN is pairs of consecutive video frames to take necessary temporal information into account. As shown by the lower part of Fig. 3, the bottom layers of our CNN are conventional CNN layers that extract spatio-temporal visual features for describing the contents of the pair of frames. Since the max pooling layers decrease spatial resolutions of the input, learnable deconvolutional layers "deconv4"-"deconv6" are added to "upsample" the feature maps of decreased resolutions. In the topmost part of the network, a convolution layer with 1×1 kernels ("conv_o") act as a regressor to output the estimated density, flow, or counting values. Similar to the GoogLeNet [15], two loss short-cuts are added to "conv5_s" and "conv6_s" for directly providing training supervision to lower layers. The upper part of Fig. 3 shows the proposed learning supervision of two different training phases, which will be introduced in the following paragraphs.

We propose to train the CNN in two phases. In the first training phase, the network is trained to predict the crowd density map D_t and the crowd velocity map V_t simultaneously. These two tasks are easier than directly predicting crossing-line crowd counting map and can therefore be better trained. The loss functions for the two tasks are designed as the L_2 distances between the estimated crowd density map $\widetilde{D}_t$ and crowd velocity map $\widetilde{V}_t$, and their ground truth,

$$L_D = \sum_p \left\| \widetilde{D}_t(p) - D_t(p) \right\|^2, \tag{4}$$

$$L_V = \sum_p \left\| \widetilde{V}_t(p) - V_t(p) \right\|^2, \tag{5}$$

and the overall loss function for the first phase is then defined as

$$L_{first} = L_D + L_V. \tag{6}$$

The joint learning of the crowd density and crowd velocity maps is illustrated by the red and blue boxes in Fig. 3. Because learning the two maps are related tasks, we let them share the feature representations by the same bottom layers. By simultaneously learning the two related task, each of the two tasks can be better trained with much fewer parameters (see experiments in Sect. 4.4).

Although we could directly multiply the estimated crowd density and crowd velocity map to predict the crossing-line crowd counting map, there might be spatial mis-matches between the two maps. The multiplication of them might deviate from the desired counting map. Illustration of density and velocity mismatching is shown in Fig. 4. The reason is that there is no term to regularize the interactions between the two maps during training. In addition, we observe that the crowd counting map instead of the density and velocity maps is closer to our true objectives.

We therefore further fine-tune the trained network in the second phase with supervision of the crossing-line crowd counting map C_t, which is generated by multiplying the supervision maps of crowd density and crowd velocity respectively. As shown by the "$\otimes$" symbol in Fig. 3, the estimated density map and velocity map are multiplied to predict the crowd counting map (illustrated as the red box in Fig. 3), while the supervision signals used in the first phase are discarded. The network is then fine-tuned in an end-to-end manner by minimizing the L_2 distance between the estimated crossing-line crowd counting map $\widetilde{C}_t$ and the corresponding ground truth C_t,

$$L_{second} = \sum_p \left\| \widetilde{C}_t(p) - C_t(p) \right\|^2. \tag{7}$$

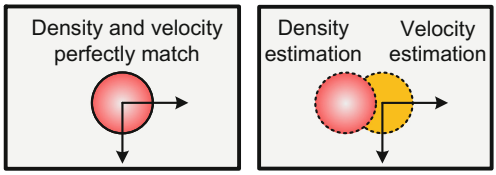


Fig. 4. (a) In an ideal case, the estimated crowd density and crowd velocity should be spatially matched. (d) Illustration of spatial mismatching between the estimated crowd density region (red) and estimated crowd velocity region (yellow). (Color figure online)

The proposed two-phase training scheme can be viewed as first dividing the original counting problem into two easier sub-problems with clear semantic meanings, *i.e.* estimating crowd density and estimating crowd velocity. Although there is no way to regularize the pairwise relations between the estimated density map and velocity map in the first phase, the trained network could serve as a good initial point for training towards the crowd counting map in the second phase, which is our desired results. In the second phase, the two maps are elementwisely multiplied and the adaptation between them are automatically optimized. Our experiments in Sect. 4.4 demonstrate that such a two-phase training strategy outperform directly learning the counting map in an end-to-end manner.

3.3 From Crowd Counting Map to LOI Counts

After our CNN model is trained, an instantaneous crossing-line crowd counting map can be predicted for every video frame by the trained counting CNN, where each entry records the estimated numbers of people passing this location along x and y directions respectively. Given an LOI, the x and y directional counting values on the LOI could be projected to its normal direction. Integrating over all the projected counting values on the LOI locations p leads to the instantaneous LOI counting numbers $c_{1,t}$ and $c_{2,t}$ in the two directions at time t,

$$c_{1,t} = \sum_{\{p|cos(\theta_p)\geq 0\}} \sqrt{C_{t,x}(p)^2 + C_{t,y}(p)^2} \cdot \cos(\theta_p), \tag{8}$$

$$c_{2,t} = \sum_{\{p|cos(\theta_p)< 0\}} \sqrt{C_{t,x}(p)^2 + C_{t,y}(p)^2} \cdot (-\cos(\theta_p)), \tag{9}$$

where θ_p is the angle between the normal direction of the LOI and the crowd counting vector $(C_{t,x}(p), C_{t,y}(p))$ at each location p. For certain period of time T, we can then integrate the instantaneous counting numbers to generate the final crossing line counts within the period as,

$$c_1 = \sum_{\{t|t\in T\}} c_{1,t}, \quad c_2 = \sum_{\{t|t\in T\}} c_{2,t}. \tag{10}$$

4 Experiments

4.1 Datasets

We contribute a new dataset for evaluating LOI crowd counting algorithms, which includes 5 different scenes, 5,900 annotated pedestrians (each one with an ID specified), and 3,100 annotated frames. A comparison with existing datasets can be seen in Table 1. As shown by the table, our dataset has more scenes and more detailed annotations (pedestrian locations vs. single counting numbers) than any existing dataset. Since we provide complete trajectory annotations for

Table 1. Statistics of existing crossing-line crowd counting datasets. [a]This dataset is not publicly available at the time of publication.

Dataset	Proposed	UCSD [3]	LHI [5]	TS-CNN[a] [2]
# of scenes	5	1	5	7
Data type	video	video	video	temporal slices
Annotation type	trajectory	trajectory	none	total counts
# of annotations	5,900	153	0	n/a
Frame resolution	1280 × 720	238 × 158	352 × 288	n/a

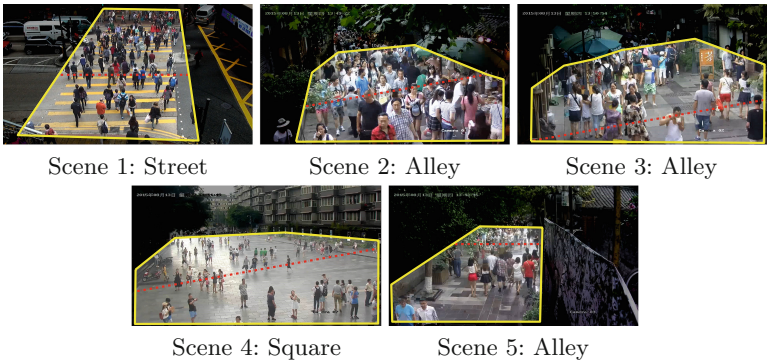


Fig. 5. Illustration of the 5 scenes and the LOIs in our datasets.

each pedestrian in our dataset, we can generate ground truth counts for arbitrary LOIs. We manually defined 1 LOI perpendicular to the traffic flow direction for each scene. Some example frames and LOIs from the 5 scenes are shown in Fig. 5. The training set is created as the first 70 % continuous frames of the 5 scenes, and the remaining 30 % frames are used as the test set. We also evaluate our proposed CNN model on the UCSD crossing-line counting dataset [10].

4.2 Experimental Setup

We compared our proposed method with the temporal-slice-based CNN model in [2] and the integer programming based method [10]. For our proposed CNN model, we randomly cropped video frame patches of size 224×224 as our training samples. Our CNN model was randomly initialized and trained with the training set of our proposed dataset. The network parameters were optimized by Stochastic Gradient Descent (SGD) with a batch size of 96. We gradually decreased the learning rate until the training is converged. During testing, pairs of whole video frames were input into our CNN to estimate the crowd counting maps. On an NVIDIA TITAN GPU, our proposed CNN model achieves a near real-time processing speed of 10 frames per second. For the temporal-slice-based CNN method [2], we generated training temporal slices of 12 s from our dataset

at the pre-defined LOIs, which result in a total of more than 50,000 temporal slices. We implemented their 3 networks following the description in [2].

To evaluate the accuracy of LOI crowd counting methods, we divided the test frames into short video clips of 10 s. The accuracy of crossing-line crowd counting was then calculated as the Mean Windowed Relative Absolute Errors (MWRAE) between the ground truth cumulative counts and the estimated cumulative counts for all short clips:

$$MWRAE = \frac{1}{N}\sum_{i=1}^{N}\frac{\|u_i - \widetilde{u}_i\|}{u_i} \times 100\,\%, \tag{11}$$

where u_i and $\widetilde{u}_i$ represent the ground truth and estimated cumulative crowd counts of ith test video clip, and N is the total number of test clips. The reason we chose this metric instead of the Mean Absolute Errors (MAE) and Mean Windowed Absolute Errors (MWAE) in existing literatures [10] is because the MAE would be sensitive to one single frame's large error while the MWAE favors video clips with fewer pedestrians. They are not comparable across different scenes and provide no information on the difficulties of different scenes. Therefore, we argue the MWRAE is a more reasonable evaluation metric for crossing-line people counting. However, when evaluating our proposed model's performance on the UCSD dataset [10], we still report the MAE and MWAE to make fair comparison with existing methods.

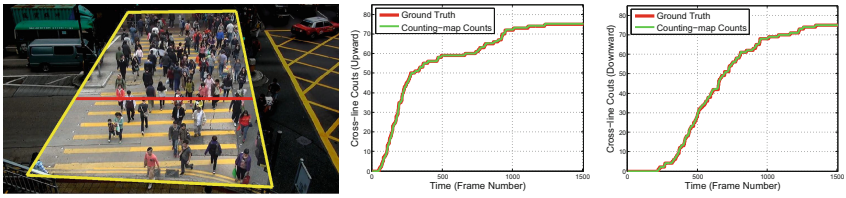
In addition to the methods from existing literature, we also designed multiple baseline models for comparison, which investigate the effects of different components of our proposed algorithm and other plausible solutions. (1) *Phase I*: this baseline network has the same structure as our proposed one but is only trained with phase I learning objectives, *i.e.*, the crowd density map and crowd velocity map. The crowd counting map is obtained by elementwisely multiplying the estimated density and velocity maps. (2) *Direct-training-A*: instead of training the proposed CNN using the proposed two-phase scheme with different learning objectives at each stage, this CNN was directly trained towards the crowd counting map. The short-cut loss signals were also modified so that they back-propagate from the crowd counting map to "conv5_s" and "conv6_ s". (3) *Direct-training-B*: this network is the same as *Direct-training-A* but "conv_o" and "deconv_o" directly output the 2-channel crowd counting map without the two elementwise multiplication branches. (4) *Two-separate*: two separated CNN models of the same structure as our proposed one were trained for estimating the crowd density map and the crowd velocity respectively. The counting map is obtained by multiplying the results of the two CNNs.

4.3 Evaluation on the Accuracy of Supervision Crowd Counting Maps

Since we generate the pixel-level crowd counting map by elementwisely multiplying a crowd density map with Gaussian density functions and a crowd velocity

Table 2. The MWRAE (%) between the ground truth crowd counts and the counts calculated by performing count mapping in Sect. 3.3 on our supervision crowd counting maps.

Scene 1	Scene 2	Scene 3	Scene 4	Scene 5	**All**
Down/Up	Down/Up	Down/Up	Down/Up	Down/Up	Down/Up
0.82/1.03	2.68/2.36	1.67/2.31	2.49/2.90	4.16/3.00	2.47/2.25



(a) Scene 1 (b) Upward counts (c) Downward counts

Fig. 6. (a) Scene 1 with its LOI. (b-c) The ground-truth LOI cumulative counts (red) and the LOI cumulative counts from our supervision counting maps (green) for the LOI in (a). (Color figure online)

map, we evaluated if such supervision counting map provides accurate approximations for calculating the correct counting numbers with our proposed evaluation metric (see Table 2). For each LOI in our dataset, the crowd counts from our supervision counting maps are generated as introduced in Sect. 3.3. The MWRAEs between the ground truth counts and counts from our counting map for all scenes are reported in Table 2. The small MWRAEs show the way of generating crowd counting maps is able to obtain accurate enough crowd counting numbers for different LOIs and the generated supervision maps are suitable for training our CNN model. In Fig. 6, we show one example LOI's ground truth crowd counts and the counts from our supervision counting maps. The count curves from our counting map closely follow those of the ground truth.

4.4 Results on the Proposed Dataset

For our proposed dataset, we evaluated the MWRAE of the cumulative counts by our proposed CNN and other compared methods. The results are reported in Table 3, which show that our proposed CNN achieves the lowest errors for most scenes.

The TS-CNN [2] is based on temporal slices of both frames and optical flow maps. Since our dataset has many crowded scenes, the temporal slices are heavily degenerated (see Fig. 1(b) for examples). TS-CNN cannot output satisfactory crowd counts based on these temporal slices. The IP-based method [10] performs worst on this dataset, because it is based on hand-crafted features and cannot effectively adapt to such complex scenes.

Table 3. MWRAE (%) by our proposed method and other compared ones on the proposed dataset.

Method	Proposed	TS-CNN [2]	IP-based [10]	*Phase I*	*Direct-A*	*Direct-B*	*Two-separate*
	Down/Up	Down/Up	Down/Up	Down/Up	Down/Up	Down/Up	Down/Up
Scene 1	**4.01**/**5.59**	55.9/53.2	479/895	5.17/6.99	5.11/10.1	12.4/22.6	4.84/8.06
Scene 2	**7.21**/14.6	17.9/31.6	398/569	10.3/18.5	7.75/13.9	7.39/**13.4**	11.5/19.1
Scene 3	**4.82**/**5.82**	19.4/23.3	505/471	6.19/9.36	13.3/8.57	6.25/7.83	6.58/10.8
Scene 4	13.4/15.1	20.3/33.5	254/362	18.5/14.9	**11.8**/**12.4**	12.9/ 14.6	20.2/16.6
Scene 5	20.7/**13.7**	**17.8**/28.0	518/646	24.4/16.7	22.5/19.3	22.7/17.7	27.4/21.7
All	**11.2**/**10.7**	29.5/36.0	454/648	14.0/13.0	13.0/13.7	14.0/17.7	15.3/15.3

For the baseline deep CNN models, the (1) *Phase I* model was trained only with phase I learning objective and was not finetuned with the final crowd counting map. This model has lower accuracy than our CNN model trained with the proposed two-phase training scheme, which demonstrates the necessity of finetuning our CNN with the phase II objective. The (2) *Direct-training-A* model learns to directly output the two-channel crowd counting map by training with only the phase II learning objectives. The network structure remains the same as that of the proposed network. The accuracy of this training scheme is lower than that of our two-phase training strategy. Since our strategy decomposes the original problem into two easier sub-problems with clear semantic meanings, the bottom layers can learn more powerful feature representations. The (3) *Direct-training-B* model also directly outputs the two-channel crowd counting map but without the elementwise multiplication layer in the top layer. It actually generates better result than the *Direct-training-A* model, which demonstrates that it is our proposed two-phase training strategy instead of the elementwise multiplication operation that contributes to the increase of the counting accuracy. For the (4) *Two-separate* model that learns two independent networks for predicting crowd density and crowd velocity separately, we compare it with our *Phase I* model, which shows that the jointly learning crowd density and crowd velocity is able to assist the learning of both tasks and results in accurate results. Note that the *Two-separate* model has twice the parameters compared with *Phase I* model and is also twice slower for evaluation.

4.5 Results on the UCSD Dataset

We also evaluated our proposed method on the UCSD dataset [3], which contains only 1 video, and the results are reported in Table 4. The IP-based method [10] based on temporal slices has very good performance on this dataset because the video has relatively low crowd density (Fig. 7), and the training and testing samples are all from the same video. Because of the small number of the training frames in the dataset, our proposed CNN model was first trained with our proposed dataset and then fine-tuned on the UCSD dataset. The size of the video in the dataset is four times smaller than the ones in our training set. We enlarge the video frames and then input into our network for evaluation. Although with

Table 4. The results on the UCSD dataset by our proposed method and the IP-based method [10].

Method	Left		Right	
	MAE	MWAE	MAE	MWAE
Proposed CNN	1.1833	0.5964	0.6285	0.4719
IP-based [10]	0.6040	0.7231	0.6883	0.5105

(a) (b)

Fig. 7. (a) Example frame of the UCSD dataset and the pre-defined LOI for the dataset. (b) Example temporal slice generated from the LOI. The shape and appearance of pedestrians could be well recognized because of the low crowd density of the dataset.

such difficulties, our proposed CNN show better accuracy compared with the integer-programming-based method in [10].

5 Conclusion

The problem of counting people crossing a line-of-interest is of great interest to the intelligent surveillance industry. In this paper, we present a deep Convolutional Neural Network for solving this problem. We observe that temporal slices used by state-of-the-art methods are sensitive to high crowd density, slow walking speed and different orientations of the LOI. A CNN model is therefore proposed to directly process pairs of video frames and predict pixel-level crossing-line crowd counting map. The proposed CNN model is trained with counting maps with rich supervision information and a novel two-phase training scheme. Such a training scheme decomposes the original problem into two easier ones with clear semantic meanings, which helps the CNN learn more discriminative feature representations. A new dataset for evaluating crossing-line crowd counting is proposed and would benefit related research along this direction.

Acknowledgments. This work is partially supported by the General Research Fund sponsored by the Research Grants Council of Hong Kong (Project Nos. CUHK14206114, CUHK14205615, CUHK419412, CUHK14203015), the Hong Kong Innovation and Technology Support Programme (No. ITS/221/13FP), National Natural Science Foundation of China (Nos. 61371192, 61301269), and Ph.D programs foundation of China (No. 20130185120039).

References

1. Bae, S.H., Yoon, K.J.: Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In: IEEE Conference Computer Vision and Pattern Recognition (2014)
2. Cao, L., Zhang, X., Ren, W., Huang, K.: Large scale crowd analysis based on convolutional neural network. Pattern Recogn. **48**(10), 3016–3024 (2015)
3. Chan, A.B., Liang, Z.S.J., Vasconcelos, N.: Privacy preserving crowd monitoring: Counting people without people models or tracking. In: IEEE Conference Computer Vision and Pattern Recognition (2008)
4. Chan, A.B., Vasconcelos, N.: Bayesian poisson regression for crowd counting. In: International Conference Computer Vision (2009)
5. Cong, Y., Gong, H., Zhu, S.C., Tang, Y.: Flow mosaicking: real-time pedestrian counting without scene-specific learning. In: IEEE Conference Computer Vision and Pattern Recognition (2009)
6. Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., van der Smagt, P., Cremers, D., Brox, T.: FlowNet: learning optical flow with convolutional networks. In: International Conference Computer Vision (2015)
7. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference Computer Vision and Pattern Recognition (2014)
8. Lempitsky, V., Zisserman, A.: Learning to count objects in images. In: International Conference Advances in Neural Information Processing Systems (2010)
9. Li, W., Zhao, R., Xiao, T., Wang, X.: Deep-reID: Deep filter pairing neural network for person re-identification. In: IEEE Conference Computer Vision and Pattern Recognition (2014)
10. Ma, Z., Chan, A.: Counting people crossing a line using integer programming and local features. IEEE Trans. Circuits Syst. Video Technol. **PP**(99), 1 (2016)
11. Milan, A., Leal-Taixé, L., Schindler, K., Reid, I.: Joint tracking and segmentation of multiple targets. In: IEEE Conference Computer Vision and Pattern Recognition (2015)
12. Possegger, H., Mauthner, T., Roth, P.M., Bischof, H.: Occlusion geodesics for online multi-object tracking. In: IEEE Conference Computer Vision and Pattern Recognition (2014)
13. Ryan, D., Denman, S., Fookes, C., Sridharan, S.: Crowd counting using multiple local features. In: International Confernce Digital Image Computing Techniques and Applications (2009)
14. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: International Confernce Learning Representations (2014)
15. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: IEEE Confernce Computer Vision and Pattern Recognition (2015)
16. Tian, Y., Luo, P., Wang, X., Tang, X.: Pedestrian detection aided by deep learning semantic tasks. In: IEEE Conference Computer Vision and Pattern Recognition (2015)
17. Wang, L., Ouyang, W., Wang, X., Lu, H.: Visual tracking with fully convolutional networks. In: International Conference Computer Vision (2015)
18. Zhang, C., Li, H., Wang, X., Yang, X.: Cross-scene crowd counting via deep convolutional neural networks. In: IEEE Conference Computer Vision and Pattern Recognition (2015)