# Crowd Detection Using Deep Learning

**Sayali Bodake[1], Dr. S.S. kadam[2]**

*[1]Student, Dept. of Technology, SPPU, Maharashtra, India*
*[2]ESEG Group Centre for Development Advanced Computing, Panchawati Rd., Panchawati, Pashan, Pune, Maharashtra, India*

---------------------------------------------------------------------***---------------------------------------------------------------------

**Abstract -** *This way of life makes life easier for people and increases the use of public services in cities. We present a CNN-MRF-based method for counting people in still images from various scenes. Crowd density is well represented by the features derived from the CNN model trained for other computer vision tasks. The neighboring local counts are strongly correlated when using the overlapping patches separated strategies. The MRF may use this connection to smooth adjacent local counts for a more accurate overall count. We divide the dense crowd visible image into overlapping patches, and then extract features from each patch image using a deep convolutional neural network, followed by a completely connected neural network to regress the local patch crowd count. Since the local patches overlap, there is a strong connection between the crowd counts of neighboring patches. We smooth the counting effects of the local patches using this connection and the Markov random field.*

***Key Words***: *Convolutional Neural Network (CNN),* Image process, Feature extraction, Feature selection, detection, classification.

## 1. INTRODUCTION

There are two major groups of existing models for estimating crowd density and counting the crowd: direct and indirect approaches. The direct approach (also known as object detection based) is based on detecting and segmenting each person in a crowd scene to get a total count, while the indirect approach (also known as feature based) takes a picture as a whole and extracts some features before getting the final count. Due to variations in perspective and scene, the distribution of crowd density in crowded crowd images is seldom consistent. Figure 3 shows several examples of photographs. As a result, counting the crowd by looking at the entire picture is irrational. As a result, the divide-count-sum approach was adapted in our system. After dividing the images into patches, a regression model is used to map the image patch to the local count. Finally, the cumulative number of these patches is used to calculate the global image count. There are two benefits of image segmentation: To begin with, the crowd density in the small picture patches has a fairly uniform distribution. Second, image segmentation improves the amount of training data available to the regression model. Because of the benefits mentioned above, we can train a more robust regression model.

Signal processing, image processing, and computer vision are only a few of the applications where CNNs come in handy. Several CNN-CC algorithms have been proposed to deal with major issues such as occlusion, low visibility, inter- and intra-object variance, and scale variation due to different viewpoints in this regard. Figure 1 depicts a typical CNN-CC flow diagram that depicts two approaches. Except for the last two blocks, which were used for comparison and error computation, the first, on the left, found ground-truth density (GTD). On the right, the second computed ED and crowd counting. The description of each block is as follows Density estimation: It is a method for estimating the probability density function of a random variable using observed (ground-truth) data. The ED of a crowd can be obtained in a variety of ways, including clustering, identification, and regression. In sparse crowds, detection-based techniques perform well, while regression-based approaches perform well in dense crowds and overestimate crowds in sparse patches. In both sparse and dense cases, a combination of detection and regression can be used to improve results.

Counting: It's a tool for counting the number of items (people, cells, vehicles, and so on) in an image or video that's used after a density map has been computed. Image density affects how well various well-known handcrafted techniques work. Counting by detection, for example, works better in sparse-density images since there are less overlapping artifacts, where as CNN-based methods work well in images with a wide density spectrum. Complex network architecture, increased number of parameters, high computational cost, and real-time deployment are some of the unique challenges faced by CNN-CC algorithms. Traditional handcrafted crowd-counting algorithms can be used for real- time monitoring, but they have lower precision and produce a low-resolution density map. In high occlusion, a wide density spectrum, and scale-varying conditions, these techniques often struggle to produce the desired results. In terms of prediction accuracy and resolution, CNN-CC algorithms, on the other hand, outperform. The cost of computation is lower in traditional handcrafted methods. The majority of applications strive for a high level of prediction precision. Many researchers attempted to reduce uncertainty and succeeded. As a result of the growing popularity of CNN-CC techniques, we decided to review and evaluate the most recent and well-known research papers on the most difficult datasets.

## 2. THEORY AND LITERATURE SURVEY

### 2.1 Algorithm Design

```
Input: Training dataset TrainData[], various activation functions[], Threshold Th

Output: Extracted Features Feature_set[] for completed trained module.

Step 1: Set input block of data d[], activation function, epoch size,

Step 2 : Features.pkl ← ExtractFeatures(d[])

Step 3 : Feature_set[] ← optimized(Features.pkl)

Step 4 : Return Feature_set[]

Test Module

Input: Train_Feature set [], // Set of training dataset

Test_Feature set [] //Set of test dataset

        Threshold denominator Th

        Collection List cL

Output: classified all instances with the desired weight.

Step 1: Read all features from the Testing dataset using the below function

        $\text{Test\_Feature} = \sum_{j=1}^{n} (\text{T}[\text{j}])$

Step 2: Read all features from the training dataset using the below function

        $\text{Train\_Feature} = \sum_{k=1}^{m} (\text{T}[\text{k}])$

Step 3: Read all features from Trainset using below

Step 4: Generate the weight of both features set

    $Weight = classifyInstance(\text{Train\_Feature, Test\_Feature})$

Step 5: Verify with Th

Selected_Instance= result = Weight >Th ? 1 : 0;

Add each selected_instance into cL, when n = null

Step 6: Return cL
```

### 2.2 Literature survey

Crowd safety in public places has always been a serious but difficult issue, especially in high-density gathering areas. The higher the crowd level, the easier it is to lose control [1], which can result in severe casualties. In order to aid in mitigation and decision-making, it is important to search out an intelligent form of crowd analysis in public areas. Crowd counting and density estimation are valuable components of crowd analysis [2], since they can help measure the importance of activities and provide appropriate staff with information to aid decision-making. As a result, crowd counting and density estimation have become hot topics in the security sector, with applications ranging from video surveillance to traffic control to public safety and urban planning [3]. A crowd monitoring system is in very high demand these days. However, current crowd monitoring system products have a number of flaws, such as being constrained by application scenes or having low precision. In particular, there is a lack of research on tracking the number of pedestrians in a large-scale crowded area [4]. The detection-based methods and the regression-based methods are the two types of crowd counting methods. Detection-based crowd counting methods typically employ a sliding window to detect each pedestrian in the scene, calculate the pedestrian's approximate location, and then count the number of pedestrians [5–7]. For low-density crowd scenes, detection-based methods may produce decent results, but they are severely restricted for high-density crowd scenes. The early regression- based methods [8–10] attempt to learn a direct mapping between low-level features derived from local image blocks and head count. Direct regression-based approaches like these only count the number of pedestrians while missing essential spatial in- formation. Learning the linear or non-linear mapping between local block features and their corresponding target density maps, as indicated by references [11,12], may integrate spatial information into the learning process. Researchers were inspired by the Convolutional Neural Network's (CNN) performance in many computer vision tasks to use CNN to learn nonlinear functions from crowd images to density maps or counts. In 20205, Wang et al. [13] used the Alexnet network structure [14] to apply CNN to the crowd counting mission. To count the number of pedestrians in the crowd picture, the completely connected layer with 4096 neurons was replaced by a layer with only one neuron.

In the same year, Zhang et al. [4] discovered that when existing approaches were applied to new scenes that varied from the training dataset, their output was significantly reduced. To address this problem, a data-driven approach was proposed for fine-tuning the pre-trained CNN model with training samples that were close to the density level in the new scenario, allowing it to adjust to unknown application scenes. This approach eliminates the need for retraining when the model is transformed to a new scenario, but it still necessitates a large amount of training data, and it is difficult to predict the density level of the new scene in practice. In 20206, Zhang et al. [16] proposed a multi-column convolutional neural network- based architecture (MCNN) based on the success of multi-column networks [15] in image recognition by constructing a network consisting of three columns of filters corresponding to the receptive fields with different sizes (large, medium, small) to adapt to changes in head size due to perspective effects or ima. Of column of the MCNN pre-trains all image blocks during training, then the three networks are combined for fine-tuning training. The training process is complicated, because there is a lot of redundancy in the structure.

Sam et al. [17] proposed in 20207 that the convolutional neural network for crowd counting (Switching CNN) be used to train regressions using a specific collection of training data patches based on different crowd densities in the picture. The network is made up of multiple independent CNN regressions, similar to a multi-column net- work, with the addition of a Switch classifier based on the VGG-16 [18] architecture to pick the best regression for each input block. Alternately, the Switch classifier and the independent regression are trained. Switching CNN, on the other hand, switches between regressions using the Switch classifier, which is very costly and often unreliable. Similar to Refs. [16,17], Kumaga et al. [19] suggested a hybrid neural network Mixture of CNNs in 20207, believing that a single predictor in various scene environments is insufficient to accurately predict the number of pedestrians (MoCNN). A combination of expert CNNs and a gated CNN makes up the model framework. On the basis of the context of the input picture, the appropriate expert CNN is adaptively selected. Expert CNNs estimate the image's head count in prediction, while gated CNN estimates each expert CNN's acceptable likelihood. These odds are then used as weighting factors in calculating a weighted average of all expert CNNs' head counts. Via gated CNN preparation, MoCNN not only trains numerous expert CNNs, but also learns the likelihood of each expert CNN's approximate head count. However, it can only be used for crowd counting estimation and does not have information on crowd density distribution. Tang et al. [20] proposed a low-rank and sparse-based deep-fusion convolutional neural network for crowd counting (LFCNN) that improved the accuracy of the projection from the density map to global counting by using a regression approach based on low-rank and sparse penalty.

## 3. METHODOLOGY

Due to variations in perspective and scene, the distribution of crowd density in crowded crowd images is seldom consistent. Figure 3 shows several examples of photographs. As a result, counting the crowd by looking at the entire picture is irrational. As a result, the divide-count-sum approach was adapted in our system. After dividing the images into patches, a regression model is used to map the image patch to the local count. Finally, the cumulative number of these patches is used to calculate the global image count. There are two benefits of image segmentation: To begin with, the crowd density in the small picture patches has a fairly uniform distribution. Second, image segmentation improves the amount of training data available to the regression model. Because of the benefits mentioned above, we can train a more robust regression model. The total crowd density distribution is continuous, despite the fact that the distribution of crowd density is not uniform. This means that neighboring picture patches should have identical densities. We often use overlaps to separate the image, which improves the relation between image patches. To compensate for potential image patch estimation errors and to get the overall result closer to the true density distribution, the

Markov random field is used to smooth the estimation count between overlapping image patches.

We use a completely connected neural network to learn a map from the above features to the local count, and a pre-trained deep residual network to extract features from image patches. Deep convolutional network features have been used in a variety of computer vision tasks, including image recognition, object detection, and image segmentation. This suggests that the deep convolutional network's learned features are applicable to a wide range of computer vision tasks. The representation ability of the learned features improves as the number of network layers increases. A deeper model, on the other hand, necessitates more data for preparation. Current datasets for crowd counting are insufficient to train a very deep convolutional neural network from scratch. To extract features from an image patch, we use a pre-trained deep residual network. Instead of learning unreferenced functions, their approach resolved the degradation issue by reformulating the layers as learning residual functions with reference to the layer inputs. To extract the deep features that reflect the density of the crowd, we use the residual network, which was trained on the ImageNet dataset for image classification. For every three convolution layers, this pre-trained CNN network generated a residual item, bringing the total number of layers in the network to 152. To get the 1000 dimensional features, we resize the image patches to 224 by 224 pixels as the model's input and extract the fc1000 layer's output.

Following that, the features are used to train a five-layer fully linked neural network. The input to the network is 1000-dimensional, and the network's number of neurons is 100-100-50-50-1. The local crowd count is the network's output. The completely linked neural network's learning role is to minimize the mean squared error of the training patches

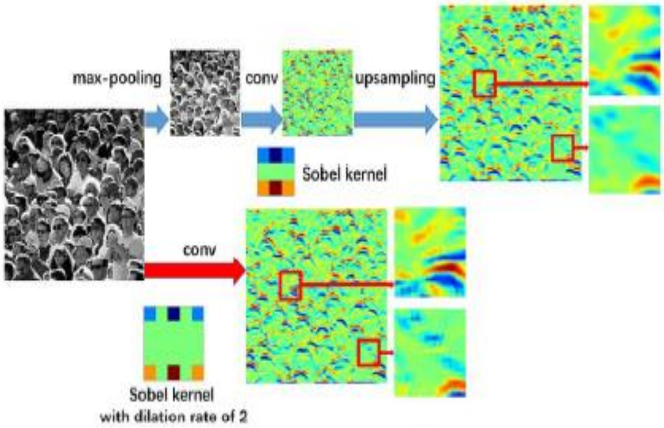


**Fig -1:** Proposed System View

**Generate Train set and Test set:** In this this phase we first create training and testing dataset for proposed system. The basic objective of this module to generate the ground truth values for both training and testing dataset.

Three different features have been extracted from each image like height, width and channel. It extracts the actual pixel values of each image during data creation. The outcome this process the .csv files both training and testing respectively.

**Pre-processing and Normalization:** Image acquisition and image resizing has used for generate fixed size of each object while Gaussian filter has used for remove the noise from object.

**CNN (Training and Testing)**: The architecture of CNN is quite different from a conventional neural network model. In the conventional neural network, input values are transformed by traversing through a series of hidden layers. Every layer is made up of a set of neurons, where each layer is fully connected to all neurons in the layer before. The reason behind the better performance of CNNs is that these networks capture the inherent properties of images. This significant feature of CNN gave us the confidence to use it in the analysis of our proposed dataset

**TensorFlow Library Module:** In the first module we implements the access interfaces and should be customized for every deep learning tools called TensorFlow. With the help of this APIs often need to be compatible with application's source code.

**Optimization:** Adam is a method that computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients *vt* like Ad delta and RMS prop, Adam also keeps an exponentially decaying average of past gradients mt, similar to momentum. Whereas momentum can be seen as a ball running down a slope, Adam behaves like a heavy ball with friction, which thus prefers flat minima in the error surface

*Mt =β1mt−1+ (1−β1)*

*Vt = β2vt−1+ (1−β2)g2t*

*mt* and *vt* are estimates of the first moment (the mean) and the second moment (the uncentered variance) of the gradients respectively, hence the name of the method. As *mt* and *vt* are initialized as vectors of 0's, the authors of Adam observe that they are biased towards zero, especially during the initial time steps, and especially when the decay rates are small.

**Mapping with ground truth:** This is the analysis module which validates the system efficiency between actual class label and predicted class label. In first modules we generate the actual count and density map each image while in testing CNN predicts the possible counts of respective input image. The ground truth score and predicted score will give accuracy of system using below formula

$$Accuracy \frac{Predcited\_Count}{GroundTrust\_counr} * 100$$

The average accuracy of system is around 85% per cent with various cross validation.

### 3.1 S/W and H/W Requirement

1. **H/W Requirement**

- Processor: CPU
- RAM: 2GB
- Hard Disk Space: 20GB
- Core: TensorFlow 1.2

2. **S/W Requirement**

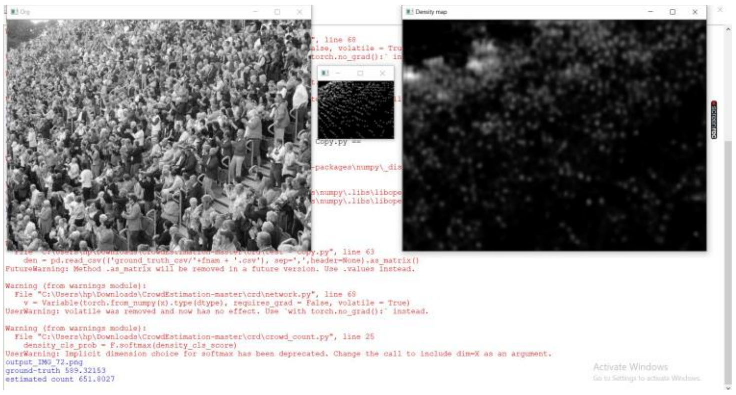
- Language: Python
- IDE: Python 3.6 and Matlab 16
- Database: MYSQL
- Platform: Microsoft Windows 10

### 3.2 Datasets Used

- **Input dataset:**
  Images that contains large number of crowd.
- **Outcomes:**
  Number of objects is available in entire image using proposed CNN approach.
- **Dataset:**
  We use some real time crowd images dataset. We also use around 5 to 10MB video that contents large number crowd. (like cricket audience video).

## 4. RESULTS AND DISCUSSION

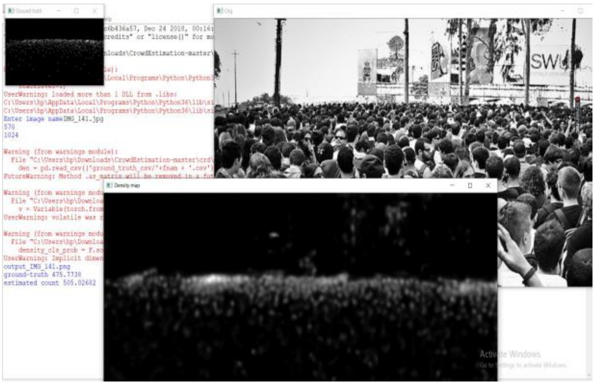
This system's work can only be measured by contrasting it to other systems that are attempting to solve a similar end-user problem. Figure.2 shows the efficiency of the proposed system with other recent approaches in literature.



```
", line 68
alse, volatile = Tru
torch.no_grad():` in
Copy.py ==
-packages\numpy\_dis
s\numpy\.libs\libope
s\numpy\.libs\libope
    copy.py", line 63
    den = pd.read_csv(('ground_truth_csv/'+fnam + '.csv'), sep=',',header=None).as_matrix()
FutureWarning: Method .as_matrix will be removed in a future version. Use .values instead.

Warning (from warnings module):
  File "C:\Users\hp\Downloads\CrowdEstimation-master\crd\network.py", line 69
    v = Variable(torch.from_numpy(x).type(dtype), requires_grad = False, volatile = True)
UserWarning: volatile was removed and now has no effect. Use `with torch.no_grad():` instead.

Warning (from warnings module):
  File "C:\Users\hp\Downloads\CrowdEstimation-master\crd\crowd_count.py", line 25
    density_cls_prob = F.softmax(density_cls_score)
UserWarning: Implicit dimension choice for softmax has been deprecated. Change the call to include dim=X as an argument.
output_IMG_72.png
ground-truth 589.32153
estimated count 651.8027
```

**Fig -2:** Experiment 1

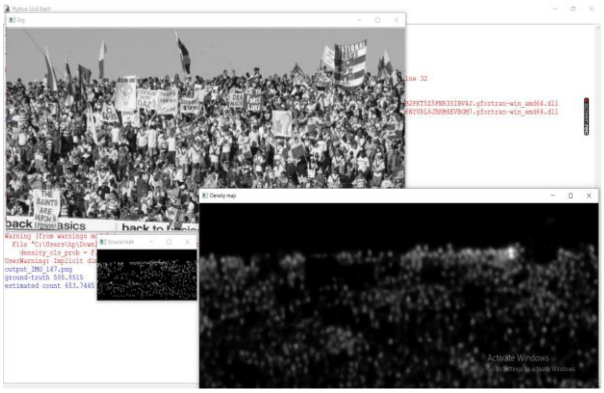
Its Ground truth count is 651, Estimated count is 589, Accuracy is 90.47% and Error Rate is 9.53%



```
c6b436a57, Dec 24 2018, 00:16:
credits" or "license()" for mo
nloads\CrowdEstimation-master\
le]:
Local\Programs\Python\Python3
UserWarning: loaded more than 1 DLL from .libs:
C:\Users\hp\AppData\Local\Programs\Python\Python36\lib\si
C:\Users\hp\AppData\Local\Programs\Python\Python36\lib\si
Enter image nameIMG_141.jpg
578
1024

Warning (from warnings module):
  File "C:\Users\hp\Downloads\CrowdEstimation-master\crd\
    den = pd.read_csv(('ground_truth_csv/'+fnam + '.csv')
FutureWarning: Method .as_matrix will be removed in a fu

Warning (from warnings modu
  File "C:\Users\hp\Downlo
    v = Variable(torch.from
UserWarning: volatile was r

Warning (from warnings modu
  File "C:\Users\hp\Downloa
    density_cls_prob = F.so
UserWarning: Implicit dimen
output_IMG_141.png
ground-truth 475.7738
estimated count 505.02682
```

**Fig -3:** Experiment 2

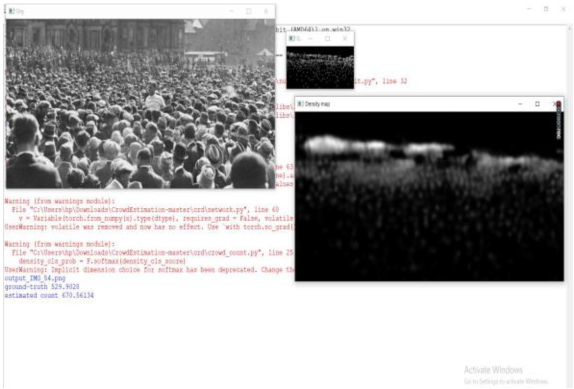
Its Ground truth count is 505, Estimated count is 475, Accuracy is 94.05% and Error Rate is 4.95%



```
line 32
B2FKT5Z5FNR3SIBV4J.gfortran-win_amd64.dll
ENYURL6JB8M4EVBGM7.gfortran-win_amd64.dll
Warning (from warnings mo
  File "C:\Users\hp\Downl
    density_cls_prob = F
UserWarning: Implicit di
output_IMG_147.png
ground-truth 585.8515
estimated count 653.7445
```

**Fig -4:** Experiment 3

Its Ground truth count is 653, Estimated count is 585, Accuracy is 89.58% and Error Rate is 9.42%



```
t.py", line 32
Warning (from warnings module):
  File "C:\Users\hp\Downloads\CrowdEstimation-master\crd\network.py", line 69
    v = Variable(torch.from_numpy(x).type(dtype), requires_grad = False, volatile
UserWarning: volatile was removed and now has no effect. Use `with torch.no_grad(

Warning (from warnings module):
  File "C:\Users\hp\Downloads\CrowdEstimation-master\crd\crowd_count.py", line 25
    density_cls_prob = F.softmax(density_cls_score)
UserWarning: Implicit dimension choice for softmax has been deprecated. Change th
output_IMG_54.png
ground-truth 529.9028
estimated count 670.56134
```

**Fig -5:** Experiment 4

Its Ground truth count is 670, Estimated count is 529, Accuracy is 78.95% and Error Rate is 21.05%

## 5. CONCLUSIONS

We present a CNN based method for counting people in still images from various scenes. Crowd density is well represented by the features derived from the CNN model trained for other computer vision tasks. The neighboring local counts are strongly correlated when using the overlapping patches separated strategies. The feature extraction may use this connection to smooth adjacent local counts for a more accurate overall count. Experimental findings show that the proposed method outperforms other recent related methods.

- The system will provide better accuracy for crowd detection from heterogeneous images.
- This approach is able to work on image as well as video dataset respectively.
- Various feature extraction selection techniques provides god detection accuracy.
- System uses RESNET from deep convolutional network that provides up to 152 hidden layers.

We can extend the system with multiple convolutional layers with ensemble deep learning model for highest accuracy.

## REFERENCES

[1] Fruin, J.J. Pedestrian Planning and Design; Metropolitan Association of Urban Designers Environmental Planners: New York, NY, USA, 1971.

[2] Zhan, B.; Monekosso, D.; Remagnino, P.; Velastin, S.; Xu, L.-Q. Crowd analysis: A survey. Mach. Vis. Appl. 2008, 19, 345–357.

[3] Zeng, L.; Xu, X.; Cai, B.; Qiu, S.; Zhang, T. Multi-scale convolutional neural net- works for crowd counting. In Proceedings of the IEEE International Conference on Image Processing, Beijing, China, 17–20 September 20207; pp. 465–469.

[4] Zhang, C.; Li, H.; Wang, X.; Yang, X. Cross-scene crowd counting via deep convolutional neural networks. In Proceedings of the IEEE Conference on Com- puter Vision and Pattern Recognition, Boston, MA, USA, 8–10 June 20205; pp. 833–841.

[5] Leibe, B.; Seemann, E.; Schiele, B. Pedestrian detection in crowded scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–26 June 2005; pp. 878–885.

[6] Zhao, T.; Nevatia, R.; Wu, B. Segmentation and tracking of multiple humans in crowded environments. IEEE Trans. Pattern Anal. Mach. Intell. 2008, 30, 1198–1211. [CrossRef] [PubMed]

[7] Ge, W.; Collins, R.T. Marked point processes for crowd counting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 2913–2920.

[8] Chan, A.B.; Liang, Z.S.J.; Vasconcelos, N. Privacy preserving crowd monitoring: Counting people without people models or tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 24–26 June 2008; pp. 1–7. Ryan, D

[9] Denman, S.; Fookes, C.; Sridharan, S. Crowd counting using multiple local fea- tures. In Proceedings of the Digital Image Computing: Techniques and Applications, Melbourne, Australia, 1–3 December 2009; pp. 81–88.

[10] Chan, A.B.; Vasconcelos, N. Bayesian poisson regression for crowd counting. In Proceedings of the IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 27 September–4 October 2009; pp. 545–551.

[11] Lempitsky, V.; Zisserman, A. Learning to count objects in images. In Proceed- ings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 6–9 December 20200; pp. 1324–1332.

[12] Pham, V.Q.; Kozakaya, T.; Yamaguchi, O.; Okada, R. Count forest: Co-voting uncertain number of targets using random forest for crowd density estimation. In Proceedings of the IEEE International Conference on

Computer Vision, Santiago, Chile, 7–13 December 20205; pp. 3253–3261.

[13] Wang, C.; Zhang, H.; Yang, L.; Liu, S.; Cao, X. Deep people counting in ex- tremely dense crowds. In Proceedings of the 23rd ACM International Conference on Multimedia, Brisbane, Australia, 26–30 October 20205; pp. 1299–1302.

[14] Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Infor- mation Processing Systems, Lake Tahoe, NV, USA, 3–6 December 20202; pp. 1097–1105.

[15] Schmidhuber, J.; Meier, U.; Ciresan, D. Multi-column deep neural networks for image classification. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 20202; pp. 3642–3649.

[16] Zhang, Y.; Zhou, D.; Chen, S.; Gao, S.; Ma, Y. Single-image crowd counting via multi-column convolutional neural network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 20206; pp. 589–597.

[17] Sam, D.B.; Surya, S.; Babu, R.V. Switching convolutional neural network for crowd counting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 20207; pp. 5744–5752.

[18] Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 20204, arXiv:1409.1556.

[19] Kumagai, S.; Hotta, K.; Kurita, T. Mixture of Counting CNNs: Adaptive Integration of CNNs Specialized to Specific Appearance for Crowd Counting. arXiv, 20207; arXiv:1703.09393.

[20] Zhang, L.; Shi, M.; Chen, Q. Crowd counting via scale-adaptive convolutional neural network. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Lake Tahoe, NV, USA, 12–14 March 20208; pp. 1113–1121.