Few-Shot Scene Adaptive Crowd Counting Using Meta-Learning

Mahesh Kumar Krishna Reddy[1] Mohammed Asiful Hossain[2] Mrigank Rochan[1] Yang Wang[1]
[1]University of Manitoba [2]Huawei Technologies Co., Ltd.
{kumarkm, mrochan, ywang}@cs.umanitoba.ca asif07hossain@gmail.com

Abstract

We consider the problem of few-shot scene adaptive crowd counting. *Given a target camera scene, our goal is to adapt a model to this specific scene with only a few labeled images of that scene. The solution to this problem has potential applications in numerous real-world scenarios, where we ideally like to deploy a crowd counting model specially adapted to a target camera. We accomplish this challenge by taking inspiration from the recently introduced learning-to-learn paradigm in the context of few-shot regime. In training, our method learns the model parameters in a way that facilitates the fast adaptation to the target scene. At test time, given a target scene with a small number of labeled data, our method quickly adapts to that scene with a few gradient updates to the learned parameters. Our extensive experimental results show that the proposed approach outperforms other alternatives in few-shot scene adaptive crowd counting.*

1. Introduction

Recently, the problem of crowd counting [16, 24, 28, 34, 35] is drawing increasing attention in computer vision research. The key reason for this surge in interest is the demand of automated complex crowd scene understanding that appears in computer vision applications such as surveillance, traffic monitoring, etc. Although the contemporary methods for crowd counting are promising, they have some significant limitations. One main limitation of existing methods is that it is hard to adapt them to a new crowd scene. This is due to the fact that these methods typically require a large number of labeled training data which is expensive and time-consuming to obtain. In this paper, we focus on this issue and propose a method that learns to adapt to a new crowd scene with very few labeled examples of that scene.

Most current approaches [16, 24, 28, 34, 35] of crowd counting treat it as a supervised regression problem where a model is learned to produce a crowd density map for the given image. In the training phase, the model learns to predict the density map of an input image given its ground-truth crowd density map as the label. The final crowd count is obtained by summing over the pixels in the estimated density map. Once the model is learned, it can be used to estimate the crowd count in test images. The main drawback of existing approaches is that they produce a single learned model that will be used in all unseen images. In order to make the model generalize well, we often need to make sure that the labeled training data is diverse enough to cover all possible scenarios which is infeasible.

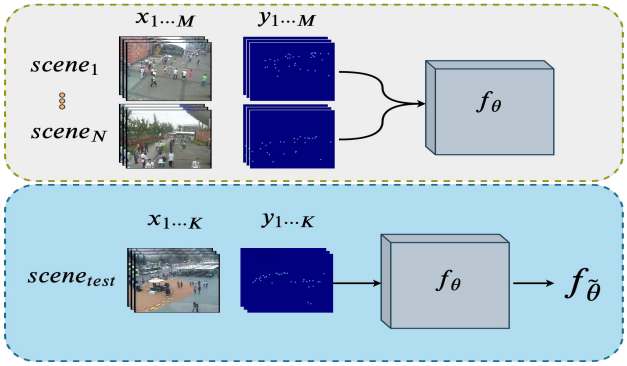


Figure 1. Illustration of our problem setting. (Top row) During training, we have access to a set of N different camera scenes where each scene comes with M labeled examples. From such training data, we learn the model parameters θ of a mapping function f_θ such that θ is generalizable across scenes in estimating the crowd count. (Bottom row) Given a test (or target) scene, we assume that we have a small number of K labeled images from this scene, where $K \ll M$ (e.g., $K \in \{1, 5\}$) to learn the scene-specific parameters $\tilde{\theta}$. With the help of meta-learning guided approach we quickly adapt f_θ to $f_{\tilde{\theta}}$ that predicts more accurate crowd count than other alternative solutions.

A recent work [10] argues that it is more effective to learn and deploy a model specifically tuned to a particular scene, instead of learning a generic model that hopefully works well in all scenes. Let us consider the video surveillance scenario. Once a surveillance camera is installed, the images captured by the camera are constrained mainly by the camera parameters and the 3D geometry of a specific scene. From the viewpoint of practical applications, we do

not need the crowd counting model to perform well on arbitrary images. Instead, we only need the model to be tuned to this particular scene. Of course, if we can get access to adequate labeled training images from this camera, a simple solution is to train a model for this scene using its training images. However, this is unrealistic since it requires collecting a large number of labeled images from the target scene whenever a new surveillance camera is installed. Moreover, generating adequate labeled data for a specific camera scene can be expensive and tedious. Ideally, we would like a way of adapting a model to work well in a new camera scene with only a few labeled examples from that scene.

We consider the few-shot scene adaptive crowd counting similar to [10]. During training, we have access to a set of training images from different scenes (e.g., each scene might correspond to one specific camera installed at one particular location). During testing, we have a new target crowd scene to which we want to adapt our model. Moreover, we consider that we have a small number (e.g., 1 or 5) of labeled images from this target scene. During training, we learn optimal (generalizable) model parameters from multiple scene-specific data by considering few-labeled images per scene. During testing, we consider the learned parameters to be a good initial point to adapt to a specific new scene. To be precise, we aim at learning the generalizable model parameters in a fashion that it produces more accurate performance when adapting to a new target scene with few gradient descent steps provided only a few labeled images from the target scene. Figure 1 shows an illustration of the problem in this paper. We address the proposed few-shot crowd counting problem using meta-learning [9] that is capable of fast adaptation to new camera scenes.

This paper makes the following contributions. First, we propose a meta-learning inspired approach to solve the few-shot scene adaptive crowd counting problem. Using the meta-learning, the model parameters are learned in a way that facilitates effective fine-tuning to a new scene with a few labeled images. Previous work in [10] uses a fine-tuning approach for this problem. The limitation of this fine-tuning approach is that it can only update certain layers that are closer to the output in the decoder to a target scene. In contrast, our approach does not have such limitation and can be used to adapt any parameters in the decoder. Second, we perform a thorough evaluation of the performance of our proposed approach on several benchmark datasets and show that the method outperforms other alternative baselines. Our approach also outperforms the fine-tuning approach in [10].

2. Related Work

Crowd Counting: The research in crowd counting can be grouped into either detection [5, 7], regression [3, 11] or density-based [15, 22] methods as proposed by [17]. Earlier work focuses on the detection and regression-based approaches. In recent years, density-based approaches using deep learning models have become popular and show superior performance. Zhang *et al*. [34] propose an approach with two learning objectives for density estimation and crowd counting. Additionally, they propose a non-parametric method to fine-tune the model to minimize the distribution difference between the source and target scenes. Zhang *et al*. [35] address crowd counting by proposing a multi-column neural network to handle an input image at multiple scales to overcome the problem of scale variations. Sam et *al*. [24] propose to estimate the density of an image patch from a regressor selected based on the density level classifier. Sindagi and Patel [28] propose to encode both local and global input image contexts to estimate the density map. In this paper, our backbone crowd counting architecture is based on [16], since it has been shown to achieve state-of-the-art performance.

In the context of crowd counting adaptation, Loy *et al*. [4] propose a non-CNN semi-supervised adaptation method by exploiting unlabeled data in the target domain. The drawback of this approach is that it requires corresponding samples that have common labels between the source and target domains. This information is usually not available in recent crowd counting datasets. Wang *et a*l. [32] propose to generate a large synthetic dataset and perform domain adaptation to the real-world target domain. One drawback of this method is that it requires the prior knowledge about the distribution of the target domain in order to manually select the scenes in the synthetic dataset. Hossain *et al.* [10] propose a one-shot adaptation approach based on fine-tuning few layers in the decoder network for adapting a crowd counting model to a specific scene.

Few-Shot Learning: The goal of few-shot learning is to learn a model from limited training examples for a task. Previously, Li *et al*. [8] propose a method for unsupervised one-shot learning by casting the problem in a probabilistic setting. Lake *et al*. [14] use compositionality and causality for one-shot scenario through Hierarchical Bayesian learning system. Luo *et al*. [18] demonstrate the transferability of representations across domains with few labeled data. A different perspective to tackle few-shot learning is by treating it is as a meta-learning problem (also known as *learning to learn* [1, 26]). The essence of using meta-learning for few-shot learning problem involves a neural network as a learner to learn about a new task with just a few instances. The recent work in meta-learning can be grouped into metric-based [13, 29, 30, 31], model-based [19, 25] or optimization-based [9, 20, 23]. The metric-based [13, 29, 30, 31] methods in general learn a distance function to measure the similarity between data points belonging to the same class. Memory or model-based [19, 25] approaches employ a memory component to

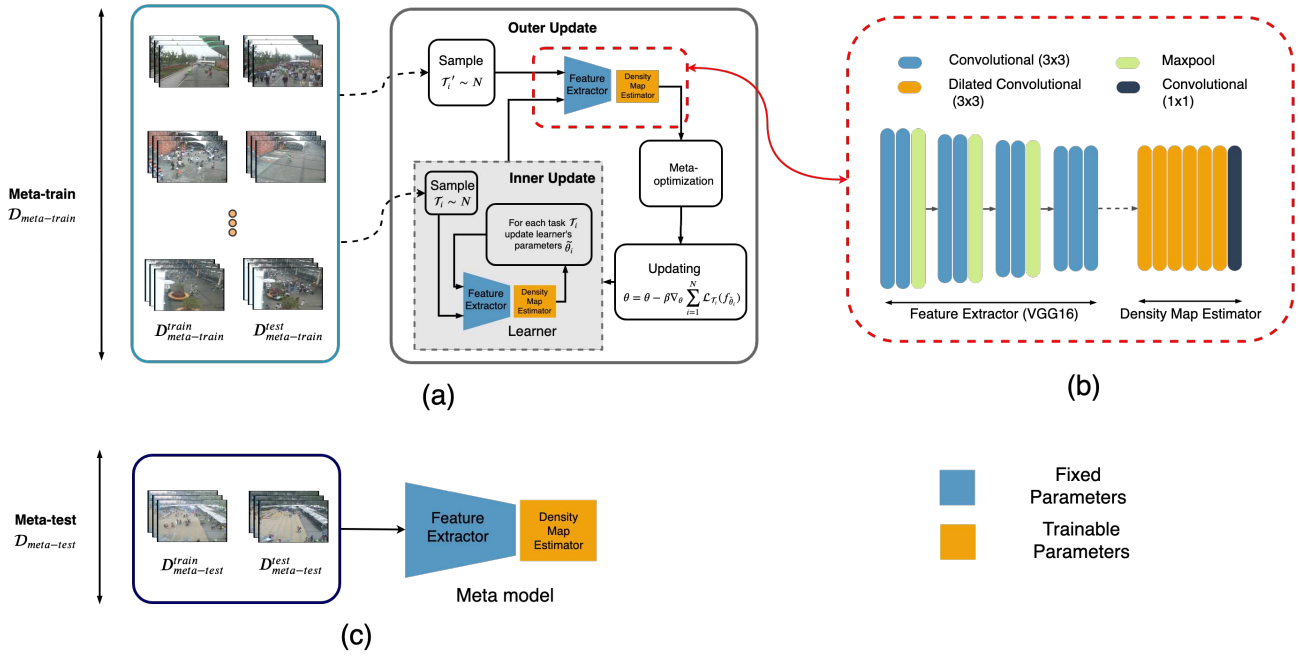


Figure 2. An overview of the main components of our model. (a) *Meta-training* stage on $\mathcal{D}_{meta-train}$. The meta-training involves optimizing an inner-update over each scene and an outer-update across different scenes. (b) Backbone crowd counting network. We use the CSRNet [16] as the backbone architecture. It comprises of a feature extractor and a density map estimator. (c) *Meta-testing* on $\mathcal{D}_{meta-test}$. We adapt the trained meta-model with θ to a new target scene by fine-tuning on K images from this scene and test on other images from this scene.

store previously used training examples. The optimization-based [9, 20, 23] frameworks learn good initialization parameters based on learning from multiple tasks that favour fast adaptation on a new task. The above works primarily target image recognition challenge, in our proposed work we follow the optimization-based meta-learning mechanism similar to [9] for a more challenging problem of crowd density estimation as it has shown to achieve superior performance compared to other optimization based methods.

3. Few-shot Scene Adaptive Crowd Counting

In this section, we first describe the problem setup for few-shot scene adaptive crowd counting (Sec. 3.1). We then introduce our proposed approach for scene adaptive crowd counting using meta-learning (Sec. 3.2).

3.1. Problem Setup

We describe how we formulate the scene adaptive crowd counting as a few-shot learning problem using meta-learning. In a traditional supervised machine learning setting, we are given a dataset $\mathcal{D} = \{D^{train}, D^{test}\}$, where D^{train} and D^{test} are the training and test sets, respectively. The goal is to learn a mapping function $f_\theta : x \rightarrow y$ that maps an input x (e.g. an input image) to its corresponding label y (e.g. the crowd density map). We use θ to denote the parameters of the mapping function f_θ. We learn θ by optimizing its corresponding loss function defined on D^{train}. After training, we test the generalization of the learned model f_θ on D^{test}.

In contrast, a few-shot meta-learning model is trained on a set of N tasks during meta-learning (*meta-training*) from $\mathcal{D}_{meta-train}$, where each task has its training and test sets. We use $\mathcal{T}_i = \{D_i^{train}, D_i^{test}\}$ $(i = 1, 2, ..., N)$, where $\mathcal{T}_i \in \mathcal{D}_{meta-train}$ to denote the i-th task (also called episode) during the meta-learning phase. The notations D_i^{train} and D_i^{test} correspond to the training set and the test set of the i-th task, respectively. Note that during the meta-learning phase, both D_i^{train} and D_i^{test} consist of labeled examples. We consider each camera scene as a task in the meta-learning formulation. Each of the training i-th scene consists of M labeled images. However, in our work, we randomly sample a small number $K \in \{1, 5\}$ and $K \ll M$ labeled images for the i-th scene in each learning iteration to form D_i^{train}. The D_i^{test} is the test set for the i-th scene. This setup reflects the real-world problem of having to learn from a few labeled images. Our goal of the meta-learning is to learn the model in a way that it can adapt to a new scene using only a few training examples from the new scene. During testing (i.e., *meta-testing*) on $\mathcal{D}_{meta-test}$, we are given a new target scene $\mathcal{T}_{new} = \{D_{new}^{train}, D_{new}^{test}\}$, where D_{new}^{train} consists of a few (e.g. K) labeled images from the target scene. The goal is to quickly adapt the

model using D_{new}^{train} so that the adapted model performs well on D_{new}^{test} which is the test data for this target scene. In our work, we use the meta-learning approach in [9] called *MAML*. MAML learns a set of initial model parameters during the meta-training stage. The model parameters learned during *meta-training* are used for initializing the model during *meta-testing* and is later fine-tuned on the few examples from a new target task. The adapted model with fine-tuned parameters is expected to perform well on the test images from the target task.

3.2. Our Approach

Consider a crowd counting model f_θ with the model parameters θ. Given an input image x, the output of $f_\theta(x)$ is a crowd density map representing the density level at different spatial locations in the image. The crowd count can be obtained by summing over entries in the generated density map. When learning to adapt to a particular scene $\mathcal{T}_i$, the model parameters are updated using a few gradient steps to optimize the loss function defined on D_i^{train}. This learning step can be considered as *inner-update* during meta-learning and the optimization is expressed as follows:

$$\begin{aligned} \tilde{\theta}_i &= \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta) \\ \text{where } \mathcal{L}_{\mathcal{T}_i}(f_\theta) &= \sum_{(x^{(j)},y^{(j)}) \in D_i^{train}} \|f_\theta(x^{(j)}) - y^{(j)}\|_F^2 \end{aligned} \tag{1}$$

where $x^{(j)}$ and $y^{(j)}$ denote a training image and its corresponding ground-truth density map from the scene $\mathcal{T}_i$, respectively. We use $||\cdot||$ to denote the Frobenius norm that measures the difference between the predicted crowd density map $f_\theta(x^{(j)})$ and the ground-truth density map $y^{(j)}$. Here α is the learning rate in the *inner-update* and its value is fixed in our implementation. We then define a loss function on D_i^{test} using $\tilde{\theta}_i$ as follows:

$$\mathcal{L}_{\mathcal{T}_i}(f_{\tilde{\theta}_i}) = \sum_{(x^{(j)},y^{(j)}) \in D_i^{test}} \|f_{\tilde{\theta}_i}(x^{(j)}) - y^{(j)}\|_F^2 \tag{2}$$

During the meta-learning phase, we learn the model parameters θ by optimizing $\mathcal{L}_{\mathcal{T}_i}(f_{\tilde{\theta}_i})$ across N different training scenes. This will effectively learn θ in a way that when we update θ with a few gradient steps in a new scene, the updated parameters $\tilde{\theta}$ will perform well on test images from this scene. This optimization problem (or *outer-update*) is similar to the optimization described in [9] and it is expressed as:

$$\theta = \theta - \beta \nabla_\theta \sum_{i=1}^{N} \mathcal{L}_{\mathcal{T}_i}(f_{\tilde{\theta}_i}) \tag{3}$$

Fig. 2 shows an illustration of this meta-learning inspired process. The result of the meta-learning phase is the set of model parameters θ. Given a new scene, we use θ to initialize the model and obtain the scene adaptive parameters $\tilde{\theta}$ by fine-tuning the parameters on the few examples from the target scene with a few gradient updates. The intuition is that a well-learned parameters θ should be able to generalize to new scenes with only a few gradient updates. In our implementation for few-shot scene adaptive crowd counting, we compute the second derivatives to optimize Eq. 3 during outer-update as described in [9].

Backbone Network Architecture: Our proposed few-shot learning approach for crowd density estimation can be used with any backbone crowd counting network architecture. In this paper, we use the CSRNet [16] (see Fig. 2) as our backbone network since it has shown to achieve state-of-the-art performance in crowd counting. The network consists of a feature extractor and a density map estimator. The feature extractor uses VGG-16 [27] to extract a feature map of the input image. Following [16], we use the first 10 layers (up to Conv4_3_3) of VGG-16 as the feature extractor. The output of the feature extractor has a resolution of $1/8$ of the input image. The density map estimator consists of a series of dilated convolutional layers [33] to regress the output crowd density map for the given image.

We use a pre-trained VGG-16 [27] model on ImageNet [6] to initialize the weights of the feature extractor part of our network. The weights of the dilated convolutional layers in the density map estimator part of the network are initialized from a Gaussian with 0.01 standard deviation. We then train the network end-to-end on the training set of WorldExpo'10 [34] dataset to learn how to produce a density map for an image containing the crowd. We refer to this trained network as "*Baseline pre-trained*" in the remaining of the paper. Note that although the baseline pre-trained model is learned on data from multiple training scenes, it is susceptible when used for adaptation in few labeled data regime as it is not specifically designed to learn from few images which we discuss in the later section. Therefore, in order to overcome this limitation, we use this baseline pre-trained network as the initialization for the meta-learning phase. During meta-learning, we fix the parameters of the feature extractor and train only density map estimator on different scene-specific data. We follow the training scheme described in this section to learn to adapt to a scene with a few labeled images.

4. Experiments

In this section, we first introduce the datasets and experiment setup (Sec. 4.1). We then describe several baselines for comparison (Sec. 4.2). We present the experimental results (Sec. 4.3).

Target	Methods	1-shot (K=1)			5-shot (K=5)		
		MAE	RMSE	MDE	MAE	RMSE	MDE
Scene 1	Baseline pre-trained	5.55	6.31	0.70	5.55	6.31	0.70
	Baseline fine-tuned	5.45 ± 0.03	6.23 ± 0.03	0.68 ± 0.004	5.06 ± 0.11	5.88 ± 0.10	0.63 ± 0.005
	Meta pre-trained	4.63	5.5	0.529	4.63	5.5	0.529
	Ours w/o ROI	3.47 ± 0.01	$\mathbf{4.19 \pm 0.01}$	0.50 ± 0.007	3.42 ± 0.03	4.81 ± 0.007	$\mathbf{0.29 \pm 0.004}$
	Ours w/ ROI	$\mathbf{3.19 \pm 0.03}$	4.30 ± 0.07	$\mathbf{0.38 \pm 0.03}$	$\mathbf{3.05 \pm 0.06}$	$\mathbf{4.19 \pm 0.15}$	0.31 ± 0.08
Scene 2	Baseline pre-trained	24.07	34.29	0.17	24.07	34.29	0.17
	Baseline fine-tuned	22.74 ± 0.47	32.92 ± 0.66	0.15 ± 0.003	20.84 ± 1.03	30.49 ± 1.37	0.156 ± 0.001
	Meta pre-trained	21.65	30.51	0.185	21.65	30.51	0.185
	Ours w/o ROI	12.05 ± 0.74	16.62 ± 1.10	0.11 ± 0.007	11.41 ± 0.54	15.35 ± 0.51	0.11 ± 0.015
	Ours w/ ROI	$\mathbf{11.17 \pm 1.01}$	$\mathbf{15.50 \pm 1.18}$	$\mathbf{0.11 \pm 0.012}$	$\mathbf{10.73 \pm 0.36}$	$\mathbf{14.95 \pm 0.60}$	$\mathbf{0.10 \pm 0.003}$
Scene 3	Baseline pre-trained	35.54	40.78	0.40	35.54	40.78	0.40
	Baseline fine-tuned	33.89 ± 0.26	39.33 ± 0.25	0.38 ± 0.03	31.05 ± 0.41	36.70 ± 0.43	0.34 ± 0.004
	Meta pre-trained	36.18	42.32	0.402	36.18	42.32	0.402
	Ours w/o ROI	8.15 ± 0.17	11.04 ± 0.42	$\mathbf{0.09 \pm 0.04}$	8.31 ± 0.54	$\mathbf{10.75 \pm 0.54}$	0.10 ± 0.009
	Ours w/ ROI	$\mathbf{8.07 \pm 0.23}$	$\mathbf{10.92 \pm 0.21}$	0.10 ± 0.007	$\mathbf{8.18 \pm 0.24}$	10.96 ± 0.31	$\mathbf{0.09 \pm 0.002}$
Scene 4	Baseline pre-trained	23.95	28.57	0.19	23.95	28.57	0.19
	Baseline fine-tuned	15.69 ± 0.28	18.96 ± 0.27	0.14 ± 0.003	16.67 ± 0.10	19.70 ± 0.16	0.15 ± 0.002
	Meta pre-trained	22.44	28.25	0.183	22.44	28.25	0.183
	Ours w/o ROI	9.74 ± 0.09	11.9 ± 0.12	0.084 ± 0.001	11.21 ± 0.47	16.1 ± 0.45	0.118 ± 0.004
	Ours w/ ROI	$\mathbf{9.39 \pm 0.26}$	$\mathbf{11.78 \pm 0.34}$	$\mathbf{0.07 \pm 0.02}$	$\mathbf{9.41 \pm 0.21}$	$\mathbf{11.91 \pm 0.17}$	$\mathbf{0.08 \pm 0.002}$
Scene 5	Baseline pre-trained	10.70	13.0	0.67	10.70	13.0	0.67
	Baseline fine-tuned	8.9 ± 0.05	11.7 ± 0.04	0.50 ± 0.03	7.79 ± 0.35	10.57 ± 0.66	0.44 ± 0.015
	Meta pre-trained	9.78	12.26	0.605	9.78	12.26	0.605
	Ours w/o ROI	4.09 ± 0.01	7.36 ± 0.01	0.196 ± 0.001	4.28 ± 0.14	7.68 ± 0.60	0.20 ± 0.001
	Ours w/ ROI	$\mathbf{3.82 \pm 0.05}$	$\mathbf{6.91 \pm 0.11}$	$\mathbf{0.192 \pm 0.001}$	$\mathbf{3.91 \pm 0.26}$	$\mathbf{7.18 \pm 0.85}$	$\mathbf{0.18 \pm 0.001}$
Average	Baseline pre-trained	19.96	24.59	0.42	19.96	24.59	0.42
	Baseline fine-tuned	17.33	21.82	0.37	16.28	20.66	0.34
	Meta pre-trained	18.93	23.76	0.38	18.93	23.76	0.38
	Ours w/o ROI	7.5	10.22	0.197	7.7	10.93	0.165
	Ours w/ ROI	**7.12**	**9.88**	**0.172**	**7.05**	**9.83**	**0.155**

Table 1. Results on WorldExpo'10 [34] test set with $K = 1$ and $K = 5$ train images in the targe scene. We report the performance our our approach with and without ROI. We also compare with three baselines *Baseline pre-trained*, *Baseline fine-tuned* and *Meta pre-trained*. We compare the results across 5 test scenes and the last two rows represent the average score for our models.

Methods	1-shot (K=1)			5-shot (K=5)		
	MAE	RMSE	MDE	MAE	RMSE	MDE
Baseline pre-trained	7.29	7.96	0.22	7.29	7.96	0.22
Baseline fine-tuned	7.11 ± 0.09	7.80 ± 0.08	0.21 ± 0.003	6.58 ± 0.07	7.32 ± 0.06	0.20 ± 0.002
Meta pre-trained	7.01	7.69	0.230	7.01	7.69	0.230
Ours w/o ROI	2.52 ± 0.08	3.26 ± 0.12	0.078 ± 0.002	2.53 ± 0.18	3.25 ± 0.27	0.078 ± 0.004
Ours w/ ROI	$\mathbf{2.44 \pm 0.02}$	$\mathbf{3.12 \pm 0.03}$	$\mathbf{0.076 \pm 0.001}$	$\mathbf{2.37 \pm 0.02}$	$\mathbf{3.04 \pm 0.01}$	$\mathbf{0.073 \pm 0.001}$

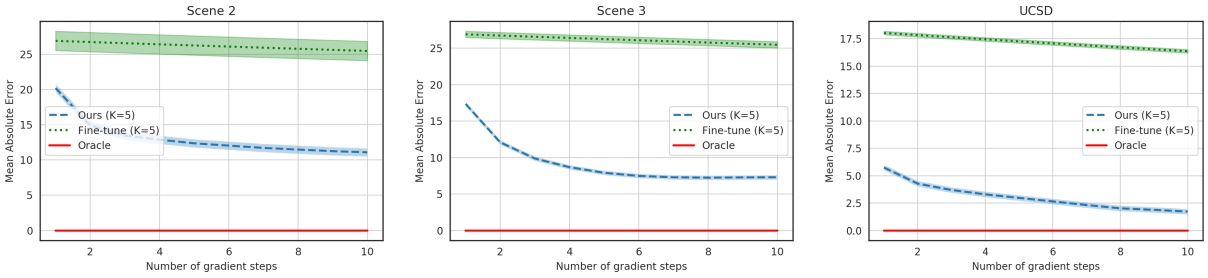
Table 2. Results on the Mall [4] dataset with $K = 1$ and $K = 5$ images in the target scene. The meta-training is performed on the WorldExpo'10 training data.

Methods	1-shot (K=1)			5-shot (K=5)		
	MAE	RMSE	MDE	MAE	RMSE	MDE
Baseline pre-trained	17.07	18.13	0.63	17.07	18.13	0.63
Baseline fine-tuned	16.41 ± 0.24	17.50 ± 0.23	0.60 ± 0.010	14.33 ± 0.16	15.55 ± 0.15	0.54 ± 0.006
Meta pre-trained	16.45	16.7	0.627	16.45	16.7	0.627
Ours w/o ROI	4.32 ± 0.74	5.57 ± 0.98	0.15 ± 0.022	3.82 ± 0.39	4.87 ± 0.58	0.14 ± 0.012
Ours w/ ROI	$\mathbf{3.08 \pm 0.13}$	$\mathbf{4.16 \pm 0.23}$	$\mathbf{0.12 \pm 0.005}$	$\mathbf{3.41 \pm 0.26}$	$\mathbf{4.22 \pm 0.36}$	$\mathbf{0.12 \pm 0.007}$

Table 3. Results on the UCSD [2] dataset with $K = 1$ and $K = 5$ images in the target scene. The meta-training is performed on the WorldExpo'10 training data.

4.1. Datasets and Setup

Datasets: Most of the available datasets for crowd-counting are not specifically designed for the scene adaptive crowd counting problem. Our problem formulation requires that the training images are from multiple scenes. To the best of our knowledge, WorldExpo'10 [34] is the only dataset with



(a) (b) (c)

Figure 3. Quantitative results of the learning curve during *meta-testing*. The graph (a) shows the learning for Scene 2 and (b) shows the result for Scene 3 in WorldExpo [34] test sets, respectively. Similarly, (c) shows the learning on UCSD [2]. Note that our approach continues to learn and achieves a lower MAE compared to the baseline fine-tuning approach in ten gradient steps. We consider $K = 5$ labeled examples in all three cases.

multiple scenes. We use this dataset for the training of our model. We also consider two other datasets (Mall [4] and UCSD [2]) for cross-dataset testing. The details of these datasets are described below.

The WorldExpo'10 [34] dataset consists of 3980 labeled images from 1132 video sequences based on 108 different scenes. We consider 103 scenes for training and the remaining 5 scenes for testing. The image resolution is fixed at 576 $\times$ 720. When testing on a target scene, we randomly choose $K \in \{1, 5\}$ images from the available images in this scene and use them for obtaining the scene adaptive model parameters $\tilde{\theta}$ (see Fig. 1). We then use the remaining images from this scene to calculate the performance of the parameters $\tilde{\theta}$. The Mall [4] dataset consists of 2000 images from the same camera setup inside a mall. The resolution of each image is 640 $\times$ 480. We follow the standard split, which consists of 800 training images and 1200 test images. Similar to the setup explained earlier, we consider $K \in \{1, 5\}$ images from the training set for fine-tuning the model to obtain the scene adaptive model parameters $\tilde{\theta}$ and later test the model on the test set. The UCSD [2] dataset consists of 2000 images from the same surveillance camera setup to capture a pedestrian scene. The crowd density is relatively sparse, ranging from 11 to 46 persons in an image. The resolution of each image is 238 $\times$ 158. We follow the standard split by considering the first 800 frames for training and 1200 images for testing. We use the same experiment setup of the Mall dataset.

Ground-truth Density Maps: All datasets come with dot annotations, where each person in the image is annotated with a single point. Following [16, 35], we use a Gaussian kernel to blur the point annotations in an image to create the ground-truth density map. We set the value of σ = 3 in the Gaussian kernel by following [16].

Implementation Details: We use PyTorch [21] for the implementation of our approach. The backbone crowd counting network is implemented based on the source code from the original CSRNet paper [16]. To generate the *Baseline pre-trained* network, we follow the procedure described in [16]. During the meta-learning phase, we initialize the network with baseline pre-trained model. We freeze the feature extractor and only train the density map estimator of the network. We set the hyper-parameters $\alpha = 0.001$ for the inner-update in SGD (see Eq. 1) and $\beta = 0.001$ in the outer-update (see Eq. 3) in Adam [12]. We randomly sample a scene for each episode during inner-update.

Evaluation Metrics : To evaluate the results, we use the standard metrics in the context of crowd count estimation. The metrics are: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and Mean Deviation Error (MDE) as expressed below:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\delta_i^{\hat{y}} - \delta_i^{y}| \tag{4}$$

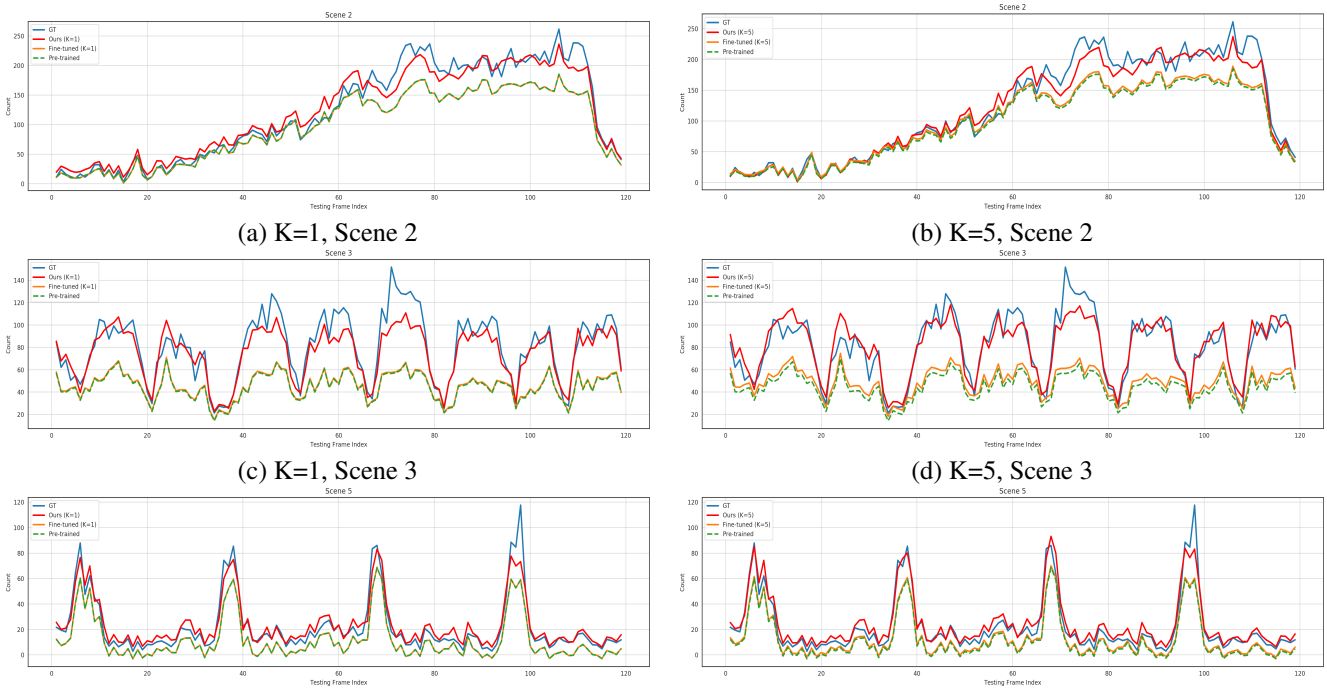
$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|\delta_i^{\hat{y}} - \delta_i^{y}|^2} \tag{5}$$

$$MDE = \frac{1}{N}\sum_{i=1}^{N}\frac{|\delta_i^{\hat{y}} - \delta_i^{y}|}{\delta_i^{y}} \tag{6}$$

where N is the total number of images in a given camera scene, $\delta_i^{\hat{y}}$ represents the crowd count of the density map generated by the model and δ_i^{y} is the corresponding crowd count of ground-truth density map for the i-th input image. Let $p_{h,w}$ be the value at the spatial location (h, w) in a density map for an image i, the count δ_i for the image can be expressed $\delta_i = \sum_{h=1}^{H}\sum_{w=1}^{W} p_{h,w}$, where $H \times W$ is the spatial size of the density map.

4.2. Baselines

We define the following baselines for comparison. Note that these baselines have the same backbone architecture as our approach.



(a) K=1, Scene 2 (b) K=5, Scene 2

(c) K=1, Scene 3 (d) K=5, Scene 3

(e)K=1, Scene 5 (f) K=5, Scene 5

Figure 4. Crowd counting performance comparison between the baselines and our approaches in different scene-specific images from WorldExpo'10 [34] dataset. The labels include, (a) $K = 1$ in Scene 2, (b) $K = 5$ in Scene 2, (c) $K = 1$ in Scene 3, (d) $K = 5$ in Scene 3, (e) $K = 1$ in Scene 5 and (f) $K = 5$ in Scene 5. Note that our approaches outperform the baselines in different settings and is robust to varying crowd density.

Baseline pre-trained: This baseline is a standard crowd counting model as in [16] trained in a standard supervised setting. The model parameters are trained from all images in the training set. Once the training is done, the model is evaluated directly on images in the new target scene without any adaptation. Note that, the original model in [16] uses the perspective maps and ground-truth ROI to enhance the final scores, we do not use them for the sake of simplicity.

Baseline fine-tuned: In this baseline, we first consider the *Baseline pre-trained* crowd counting model learned θ from the standard supervised setting. For a given new scene during testing, we fix the parameters of the feature extractor and fine-tune only the density map estimator using a few images $K \in \{1, 5\}$ from the target scene.

Meta pre-trained: This baseline is similar to our approach, but without the fine-tuning on the target scene. Intuitively, it is similar to "baseline pre-trained".

4.3. Experimental Results

Main Results: Table 1 shows the results on the WorldExpo'10 dataset for the 5 test (or target) scenes. We show the results of using both $K = 1$ and $K = 5$ images for fine-tuning in the test scene. This dataset also comes with ground-truth region-of-interest (ROI). We report the results *with* (*w/*) and *without* (*w/o*) ROI. We repeat the experiments 5 times in each setting with K randomly selected images. We average the scores across the 5 trials and report the standard deviation along with the mean of the scores in Table 1. We report the results from our models as "*Ours w/o ROI*" and "*Ours w/ ROI*". We compare with the three baselines defined in Sec. 4.2. Our models outperform the baselines in most cases. This shows that the meta-learning fine-tuning improves the model's performance. Note that our problem setup requires K labeled images in the test set and hence these K images have to be excluded in the calculation of the evaluation metrics, i.e., we have slightly fewer test images for the results in Table 1. Therefore, the performance numbers in Table 1 should not be directly compared with previously reported numbers in the crowd counting literature since our problem formulation is completely different. Besides, some previous crowd counting works [16] use additional components (e.g., perspective maps) to enhance the final performance. We do not consider these additional components in our models for the sake of simplicity (also the publicly available source code for [16] does not implement those extra components), so the number for "Baseline pre-trained" in Table 1 is slightly worse than the number reported in [16].

Table 2 and Table 3 show the results on the Mall and UCSD datasets, respectively. Here we use the training data

Target	Methods	1-shot (K=1)			5-shot (K=5)		
		MAE	RMSE	MDE	MAE	RMSE	MDE
WorldExpo (Avg.)	Meta-LSTM [23]	13.33	18.22	0.252	12.7	16.61	0.223
	Reptile [20]	11.63	15.07	0.260	8.20	11.31	0.181
	Ours w/o ROI	7.5	10.22	0.197	7.7	10.93	0.165
	Ours w/ ROI	**7.12**	**9.88**	**0.172**	**7.05**	**9.83**	**0.155**
Mall	Meta-LSTM [23]	3.95 ± 0.04	4.34 ± 0.537	0.12 ± 0.002	3.54 ± 0.44	4.41 ± 0.472	0.10 ± 0.014
	Reptile [20]	2.55 ± 0.07	3.26 ± 0.09	0.079 ± 0.001	2.49 ± 0.23	3.20 ± 0.29	0.078 ± 0.006
	Ours w/o ROI	2.52 ± 0.08	3.26 ± 0.12	0.078 ± 0.002	2.53 ± 0.18	3.25 ± 0.27	0.078 ± 0.004
	Ours w/ ROI	$\mathbf{2.44 \pm 0.02}$	$\mathbf{3.12 \pm 0.03}$	$\mathbf{0.076 \pm 0.001}$	$\mathbf{2.37 \pm 0.02}$	$\mathbf{3.04 \pm 0.01}$	$\mathbf{0.073 \pm 0.001}$
UCSD	Meta-LSTM [23]	14.15 ± 0.48	16.29 ± 0.425	0.463 ± 0.018	13.81 ± 0.10	15.99 ± 0.009	0.45 ± 0.004
	Reptile [20]	5.64 ± 2.05	6.85 ± 2.06	0.20 ± 0.075	4.48 ± 0.88	5.62 ± 0.99	0.166 ± 0.033
	Ours w/o ROI	4.32 ± 0.74	5.57 ± 0.98	0.15 ± 0.022	3.82 ± 0.39	4.87 ± 0.58	0.14 ± 0.012
	Ours w/ ROI	$\mathbf{3.08 \pm 0.13}$	$\mathbf{4.16 \pm 0.23}$	$\mathbf{0.12 \pm 0.005}$	$\mathbf{3.41 \pm 0.26}$	$\mathbf{4.22 \pm 0.36}$	$\mathbf{0.12 \pm 0.007}$

Table 4. The overall results for adaptation on WorldExpo'10 [34] test set, Mall [4] and UCSD [2] with $K = 1$ and $K = 5$ train images. We compare with other optimization based meta-learning approaches "Reptile" [20] and "Meta-LSTM" [23].

Methods	1-shot (K=1)	
	MAE	RMSE
Hossain *et al.* [10]	8.23	12.08
Ours w/o ROI	7.5	10.22
Ours w/ ROI	**7.12**	**9.88**

Table 5. Comparison of results on the WorldExpo'10 [34] dataset with $K = 1$ images in the target scene with Hossain *et al.* [10]. We use the standing train/test split on WorldExpo'10. Our approach outperforms Hossain *et al.* [10].

of WorldExpo'10 for the meta-learning. We then use Mall and UCSD for the scene adaptation and evaluation. This cross-dataset testing can demonstrate the generalization of the proposed method. Our model clearly outperforms the baselines.

To gain further insights into our method, we visualize the MAE over the number of gradient steps in Fig. 3 for different scenes. In all the cases, our proposed approach has a better start in learning and improves continuously with more gradient steps. In the three cases shown in Fig. 3, our approach performs significantly better that fine-tuning with the same number of gradient updates.

In Fig. 4, we show the comparison of the crowd count estimations between our approaches and baselines in different scenes in WordExpo'10. Our method consistently produces crowd counts that are closer to the ground-truth compared with other baselines. Some qualitative examples of the density maps generated by our method and baselines are shown in Fig. 1 of the supplementary material.

In Table 5, we compare our results with the one-shot scene adaptation proposed in [10] based on the standard WorldExpo'10 data split. In [10], the last two layers in the pre-trained model are fine-tuned to adapt to the target scene. Our approach outperforms [10].

Ablation Studies: Our approach is based on MAML [9]. In the literature, there are other optimized-based meta-learning approaches, e.g., [20, 23]. We perform additional ablation studies on the effect of different meta-learning frameworks. The results are shown in Table 4. Nichol *et al*. [20] propose an optimization based meta-learning approach similar to [9] using gradient descent. However, [20] differs from [9] in that it does not consider the second-order derivative in the meta-optimization. As a result, its performance is slightly lower as reported in Table 4 although it converges faster. In case of [23], the meta-learner is a LSTM based model unlike in [9] and [20], because of the similarity between the gradient update in backpropagation and the cell-state update in LSTM. The drawback of [23] is the large number of trainable parameters and different architectures for the learner and meta-learner. In general, the MAML-based meta learning that our approach uses outperforms other optimization-based meta-learning approaches. In Table 4, we highlight only the average score across 5 scenes in WorldExpo due to page limit. For more detailed results of every scene in WorldExpo, refer to Table 1 in the supplementary material. The training scheme for [20, 23] is similar to our approach based on the same backbone network [16] as described in the implementation details (Sec 4.1). For the hyperparameters setting, please refer to [20, 23].

5. Conclusion

In this paper, we have addressed the problem of few-shot scene adaptation for crowd counting. We have proposed a meta-learning inspired approach to address the learning mechanism for few-shot scenario. Our proposed approach learns the model parameters in a way that facilitates fast adaptation to new target scenes. Our experimental results show that our proposed approach can learn to quickly adapt to new scenes with only a small number of labeled images from the target camera scene. We believe that our work will help to increase the adoption of crowd counting techniques in the real-world applications.

Acknowledgement: We thank NSERC for funding and NVIDIA for donating some of the GPUs used in this work.

References

[1] S. Bengio, Y. Bengio, J. Cloutier, and J. Gecsei. On the optimization of a synaptic learning rule. In *Preprints Conf. Optimality in Artificial and Biological Neural Networks*, pages 6–8. Univ. of Texas, 1992.

[2] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[3] A. B. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting. In *IEEE International Conference on Computer Vision*, 2009.

[4] C. Change Loy, S. Gong, and T. Xiang. From semi-supervised to transfer counting of crowds. In *IEEE International Conference on Computer Vision*, 2013.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[7] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[8] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

[9] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 2017.

[10] M. A. Hossain, M. K. Krishna Reddy, M. Hosseinzadeh, O. Chanda, and Y. Wang. One-shot scene-specific crowd counting. In *British Machine Vision Conference*, 2019.

[11] H. Idrees, I. Saleemi, C. Seibert, and M. Shah. Multi-source multi-scale counting in extremely dense crowd images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[12] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[13] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *International Conference on Machine Learning*, 2015.

[14] B. M. Lake, R. R. Salakhutdinov, and J. Tenenbaum. One-shot learning by inverting a compositional causal process. In *Advances in Neural Information Processing Systems*, 2013.

[15] V. Lempitsky and A. Zisserman. Learning to count objects in images. In *Advances in Neural Information Processing Systems*, 2010.

[16] Y. Li, X. Zhang, and D. Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[17] C. C. Loy, K. Chen, S. Gong, and T. Xiang. Crowd counting and profiling: Methodology and evaluation. In *Modeling, simulation and visual analysis of crowds*. 2013.

[18] Z. Luo, Y. Zou, J. Hoffman, and L. F. Fei-Fei. Label efficient learning of transferable representations acrosss domains and tasks. In *Advances in Neural Information Processing Systems*, 2017.

[19] T. Munkhdalai and H. Yu. Meta networks. In *International Conference on Machine Learning*, 2017.

[20] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[21] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *Advances in Neural Information Processing Systems Workshop*, 2017.

[22] V.-Q. Pham, T. Kozakaya, O. Yamaguchi, and R. Okada. Count forest: Co-voting uncertain number of targets using random forest for crowd density estimation. In *IEEE International Conference on Computer Vision*, 2015.

[23] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

[24] D. B. Sam, S. Surya, and R. V. Babu. Switching convolutional neural network for crowd counting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[25] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning*, 2016.

[26] J. Schmidhuber. Evolutionary principles in self-referential learning. *On learning how to learn: The meta-meta-... hook.) Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1987.

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[28] V. A. Sindagi and V. M. Patel. Generating high-quality crowd density maps using contextual pyramid cnns. In *IEEE International Conference on Computer Vision*, 2017.

[29] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

[30] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[31] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, 2016.

[32] Q. Wang, J. Gao, W. Lin, and Y. Yuan. Learning from synthetic data for crowd counting in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[33] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations*, 2016.

[34] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-scene crowd counting via deep convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[35] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma. Single-image crowd counting via multi-column convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.