Domain Adaptation in Crowd Counting

Mohammad Asiful Hossain*, Mahesh Kumar Krishna Reddy†, Kevin Cannons*, Zhan Xu* and Yang Wang†
*HUAWEI Technologies Canada †University of Manitoba
Email: {mohammad.asiful.hossain, kevin.cannons, zhan.xu}@huawei.com*, {kumarkm, ywang}@cs.umanitoba.ca†

***Abstract*—We consider the problem of domain adaptation in crowd counting. Given an input image of a crowd scene, our goal is to estimate the count of people in the image. Previous work in crowd counting usually assumes that training and test images are captured by the same camera. We argue that this is not realistic in real-world applications of crowd counting. In this paper, we consider a domain adaptation setting in crowd counting where we have a *source domain* and a *target domain*. For example, these two domains might correspond to cameras at two different locations (i.e., with differing viewpoints, illumination conditions, environment objects, crowd densities, etc.). We have enough labeled training data from the source domain, but we only have either unlabeled data or a small number of labeled data in the target domain. Our goal is to train a crowd counting system that performs well in the target domain. We believe this setting is closer to real-world deployment of crowd counting systems. Due to the domain shift, a model trained from the source domain is unlikely to perform well in the target domain. In this paper, we propose several domain adaptation techniques for this problem. Our experimental results demonstrate the superior performance of our proposed approach on several benchmark datasets.**

***Keywords*-domain adaptation; crowd counting; few-shot learning; density map;**

I. Introduction

We address the problem of domain adaptation in crowd counting. Given crowd scenes from two different domains, namely a *source domain* and a *target domain*, our goal is to learn a crowd counting model that performs well in the target domain. We assume that in the source domain we have access to far more labeled examples than the target domain. For instance, this scenario may arise in the case of crowd-monitoring cameras installed at two different locations. It is very likely that the images captured from these two cameras will have a domain shift due to the fact that the respective images are going to be visually dissimilar (e.g., different viewpoints, illumination conditions, environment objects, crowd densities), as they belong to different crowd scenes. Therefore, it is essential to design efficient domain adaptation-based crowd counting systems (see Fig. I).

Crowd counting is a research area in computer vision that has many potential applications, such as traffic control and urban planning [1]. Most previous work in this area uses supervised learning. However, the supervised learning setting is not suitable for many real-world applications. To better understand this, let us consider the datasets commonly used in this area of research. For some datasets (e.g. [2], [3]) used in this area, both training and testing images are from the same camera. In this case, the learned model might overfit to the specific scene captured in the camera. This implies that in order to deploy the crowd counting system to the target camera, we require a large number of labeled training images collected by the target camera. However, this is not realistic in most applications. For some datasets (e.g. [4], [5]), both training and testing images are captured by different cameras from various scene locations. In this case, the learned model will not overfit to a particular scene and may perform better across different scenes. However, this setting has a disconnect from how crowd counting systems are deployed in real-world applications. In practice, once a camera is installed, the crowd counting system on target camera only needs to operate on images captured by this particular camera. We do not necessarily need the crowd counting system to work across different scenes and cameras. In other words, in crowd counting applications, it is often reasonable to build a system tuned to the target scene. However, the challenge is that there is typically insufficient training data from the target scene for a fully supervised approach.

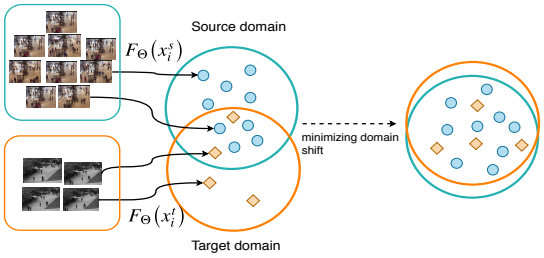

Figure 1. Illustration of high-level intuition of domain adaptation. We have a source domain and a target domain. The key idea is to minimize the domain shift between the feature representations of images from these two domains.

In this paper, we study the new problem of domain adaptation in crowd counting. We believe this new problem setup is closer to real-world applications. Consider the scenario where a company tries to develop crowd counting systems to be deployed to a user. The company might set up a camera at some location to collect labeled training images. We can consider these images to be from the *source domain*. In addition, we can consider the images captured by the user's camera to be from the *target domain*. A successful

domain adaptation solution will enable a model to work well in the target domain by taking advantage of the available training images in the source domain. In this paper, we consider three different settings of domain adaptation: 1) *unsupervised domain adaptation*, where the target domain has a large number of unlabeled images; 2) *supervised domain adaptation*, where the target domain contains a small number of labeled images; 3) *semi-supervised domain adaptation*, where the target domain consists of a small number of labeled images and a large number of unlabeled images. These three settings correspond to different application scenarios. For example, if we want to add the crowd counting capability to an existing camera that has been operating for some time, this camera would have already collected a large amount of unlabeled images in the target domain. In this case, we can use the *unsupervised domain adaptation*. In contrast, if we want to deploy a new camera at some location, we will not have to access a large amount of unlabeled images in the target scene since the camera has not be operating for a long time. In this situation, it might be reasonable to expect that the user can collected a small amount of labeled images in the target scene during the initial calibration stage after the camera was just put into operation. *Supervised domain adaptation* can be applied in this scenario. The *semi-supervised domain adaptation* is applicable when there are a large number of labeled images in the source domain and a small amount of labeled images in the target domain.

The contributions of this paper can be summarized as follows. First, this is the first paper on deep learning based domain adaptation in crowd counting. Although there exists an early work [6] on this problem, this work uses hand-crafted features. It also requires "corresponding examples" with identical labels from source and target domains. As a result, it is difficult to apply the method of [6] in general settings. Most existing deep learning based domain adaptation work focuses on tasks such as image classification, and segmentation. To the best of our knowledge, there has not been much work on domain adaptation for crowd counting. The closest work to ours is [7] which learns a model for counting different object types (e.g., people, penguins, cars). Each object type has a domain-specific layer that adjusts to that object. The main goal of [7] is different from ours. Our goal is to adapt a model that works well on a target domain with limited labeled data, while [7] focuses on allowing a model to work well on several object types simultaneously. We believe crowd counting is an area more likely to benefit from domain adaptation since the test images often come from the same camera at a fixed location and angle (i.e., one domain). Moreover, we only care how well the system works in that particular domain. Second, we propose three different domain adaptation settings in crowd counting, which can be applied in various application settings depending on specific requirements. Finally, we experimentally demonstrate that our proposed approach outperforms other alternatives. In addition to standard domain adaptation where the object of interest is the same in both domains, we also demonstrate cross-object domain adaptation. For example, in this paper the cross-object domain experiment considers people counting in the source domain and car counting in the target domain.

II. Related Work

Crowd Counting: Earlier work on crowd counting uses either detection or regression-based approaches. The detection-based approaches use region proposals [8], [9] to generate candidate regions of the crowd. The regression-based crowd estimation techniques do not consider the precise locations of people in the images. In recent years, convolutional neural networks (CNNs) have shown to achieve great success in crowd counting. The CNN-based approaches involve a learnable mapping function from the input (image) to the output space (a density map). Zhang et al. [5] propose cross-scene crowd counting with alternating objective functions during training. Zhang et al. [10] propose a Multi-column Convolutional Neural Network (MCNN) to handle perspective distortion and scale variance. Consequently, Hossain et al. [11] propose crowd counting using a scale aware attention network. Sam et al. [1] propose a Switch-CNN architecture to handle crowd density variation in an image. Sindagi and Patel [12] introduce a Contextual Pyramid CNN (CP-CNN) by incorporating the global and local contextual information from the crowd images to generate high-quality crowd estimation and density. Liu et al. [13] first learn an initial counting model by ranking different regions on unlabaled data, and then fine-tune the counting network on a smaller set of labeled data.

Few-Shot Learning: The goal of few-shot learning is to learn a visual representation from a small number of examples. Some early work in few-shot learning [14] uses Bayesian approaches for object recognition. Koch et al. [15] propose a deep Siamese neural network for similarity-based ranking between inputs and later using the trained model to generalize on new data from a different distribution with a single image. Lake et al. [16] propose a generative model based on one-shot learning. Vinyals et al. [17] propose a matching network that adopts ideas from metric learning for training a model to map from small labeled and unlabeled sets to their corresponding labels. Hossain et al. [18] propose scene specific crowd counting using the concept of one-shot learning.

Domain Adaptation: Our work is related to domain adaptation [19], [20] where the objective is to transform the models learned in one domain (i.e., source domain) such that they perform well in a in a slightly different domain (i.e., target domain). Over the years, various domain adaptation methods have been proposed. Some previous works [21], [22] focus on instance-based domain adaptation and others [23], [24],

[25], [26] focus on features domain adaptation. The instance-based approaches primarily dealt with the reduction of the discrepancy between the domains, whereas the feature-based approaches find a common latent space to match the domains.

Some domain adaptation approaches work by enforcing similarity between the features from the two domains. Gretton et al. [27] propose a statistical test based on a distance method to measure the similarity between two distributions by mapping the distributions to Reproduce Kernel Hilbert Space (RKHS) also commonly known as Maximum Mean Discrepancy (MMD). Huang et al. [28] use MMD by re-weighting training data points to minimize the domain gap.

III. Problem Setup

We assume that we have a source domain $\mathcal{D}_s$ and a target domain $\mathcal{D}_t$ in the crowd counting problem. For example, the source and target domains might correspond to images captured by two different cameras installed at two different locations (i.e., with different viewing angles, illumination conditions, scene objects, etc.). Our goal is to deploy a crowd counting system in the target domain. Let us suppose we do not have enough labeled training images in the target domain. However, we have an extensive collection of labeled images in the source domain. Due to the domain shift, a model directly trained in the source domain may not work well in the target domain. In this paper, we would like to use domain adaptation to transfer the knowledge from the source domain to the target domain.

The source domain $\mathcal{D}_s$ consists of N_s labeled images, i.e. $\mathcal{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$ where (x_i^s, y_i^s) denote the i-th image and its corresponding label in the source domain. Usually, y_i^s is represented as a density map. Depending on the application scenario and the availability of data in the target domain $\mathcal{D}_t$, we consider three domain adaptation settings defined as follows.

Unsupervised Domain Adaptation (UDA): In this setup, the target domain consists of a large collection of unlabeled images, i.e. $\mathcal{D}_t = \{x_i^t\}_{i=1}^{N_t}$ where N_t is the total number of images and x_i^t denotes the i-th image in the target domain. The unsupervised domain adaptation setup is useful in the following application scenario. Suppose the target domain corresponds to a surveillance camera that is already installed at a certain location and has been running over a period of time. In this case, we already have access to a large number of images in the target domain. Now we would like to add a new capability to this installed camera for crowd counting in the target domain. In the ideal case, we can collect labeled training images in the target domain by manually annotating the unlabeled images that have been collected by this camera. However, this is very expensive and time-consuming since it requires the end-user to annotate their images. The UDA setup will allow us to make use of unlabled images in the target domain that are readily available.

Supervised Domain Adaptation (SDA): The UDA setup is not applicable in some real-world applications. For example, suppose the target domain corresponds to a newly installed camera. In this case, we do not have access to a large number of unlabeled images captured by this camera. However, it might be reasonable for the user to collect a small number of images and manually label them. In this setup, the target domain consists of a small amount of labeled images, i.e. $\mathcal{D}_t = \{(x_i^t, y_i^t)\}_{i=1}^{K}$ where (x_i^t, y_i^t) denote the i-th training image and its label in the target domain. Here K is the number of labeled training images in the target domain. Usually, K is very small. In this paper, we consider $K \in \{1, 5, 10\}$ in the experiments.

Semi-Supervised Domain Adaptation (SSDA): This setup is similar to UDA. But we assume that the user is able to annotate a small number of images in the target domain. In other words, the target domain $\mathcal{D}_t$ consists of two subsets of training images, i.e. $\mathcal{D}_t = \mathcal{D}_t^l \cup \mathcal{D}_t^u$. The subset $\mathcal{D}_t^l = \{(x_i^t, y_i^t)\}_{i=1}^{K}$ contains K labeled images in the target domain. Again, K is a very small number. The subset $\mathcal{D}_t^u = \{x_i^t\}_{i=K+1}^{N_t}$ contains $N_t - K$ unlabeled images. In other words, the total number of images in the target domain is N_t. Only a small subset K of them are labeled. Note that UDA and SDA can be considered as special cases of SSDA when $\mathcal{D}_t^l = \emptyset$ (i.e. $K = 0$) and when $D_t^u = \emptyset$ (i.e. $K = N_t$ and is a small number), respectively.

IV. Our Approach

We firstly introduce the backbone network, then describe our learning mechanism for domain adaptation.

A. Network Architecture

Our approach is general and can be used with any crowd counting network. Exploration of the best backbone architecture is not our focus and is beyond the scope of this paper. For simplicity, we use the multi-column convolutional neural network (MCNN) [10] as the backbone architecture in this paper, although our approach is certainly not limited to this particular choice of the backbone network. We use Θ to denote the model parameters of MCNN. For an input image x, we use $F_\Theta(x)$ to denote the final feature map before the prediction and $\Psi_\Theta(F_\Theta(x))$ to represent the predicted density map. In a supervised setting, Θ is learned by optimizing a loss function defined on labeled images.

B. Domain Adaptation for Crowd Counting

Without loss of generality, we present our domain adaptation approach in the following under the semi-supervised setting (SSDA), since the other two setups (UDA and SDA) are just special cases of SSDA.

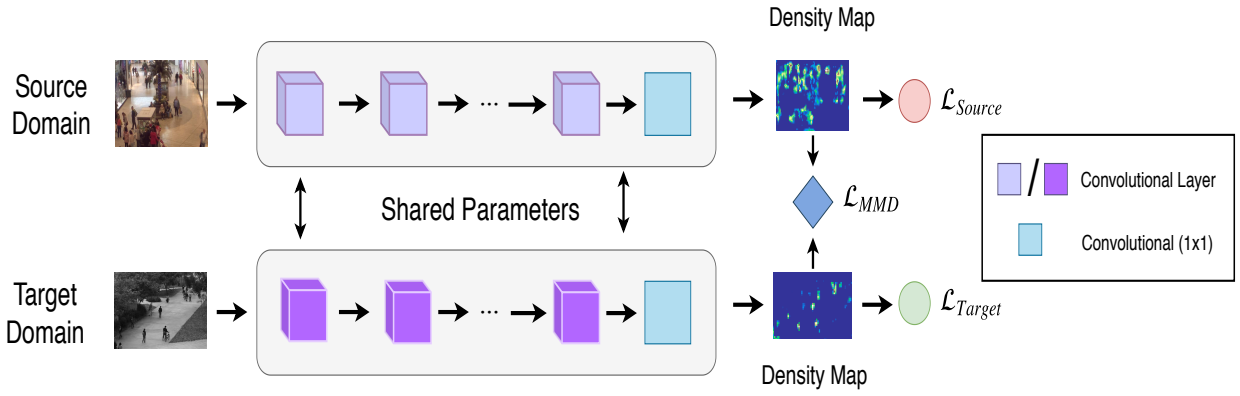

Figure 2. The architecture of our backbone network based on the multi-column convolutional neural network (MCNN) [10]. This architecture consists of a parallel branch to extract features at different scales. The features from all three branches are concatenated together in the end to predict the output. The output of the network is an estimated density map of the input image. Taking an integral over the density map gives the final count. In our work, we consider the blue colored network as source and the purple network as the target. We employ maximum mean discrepancy to minimize the domain shift between the source and target feature representations.

In the SSDA setting, we learn the model parameters Θ of MCNN by optimizing the following loss functions:

$$\mathcal{L}_{SSDA}(\Theta) = \mathcal{L}_{source}(\Theta) + \mathcal{L}_{target}(\Theta) + \alpha\, \mathcal{L}_{mmd}(\Theta) \quad (1)$$

where $\mathcal{L}_{source}(\Theta)$ and $\mathcal{L}_{target}(\Theta)$ in Eq. 1 are loss functions defined on $\mathcal{D}_s$ and $\mathcal{D}_d^l$, respectively. The last term $\mathcal{L}_{mmd}(\Theta)$ is a loss defined for domain adaptation. The hyperparameter α controls the relative weight of the domain adaptation loss. The details of these loss functions are described in the following.

Supervised Loss: The loss $\mathcal{L}_{source}(\Theta)$ measures how well the model parameters perform on the labeled training images in the source domain $\mathcal{D}_s$. It is defined as:

$$\mathcal{L}_{source}(\Theta) = \frac{1}{N_s} \sum_{i=1}^{N_s} ||F_\Theta(x_i^s) - y_i^s||_F^2 \quad (2)$$

where y_i^S denotes the ground-truth density map of the x_i and $||\cdot||_F$ denotes the Frobenius norm.

Similarly, the loss $\mathcal{L}_{target}(\Theta)$ measures how well the model parameters perform on the small set of labeled training images in the target domain $\mathcal{D}_t^l$. It is defined as:

$$\mathcal{L}_{target}(\Theta) = \frac{1}{K} \sum_{i=1}^{K} ||F_\Theta(x_i^t) - y_i^t||_F^2 \quad (3)$$

Domain Adaptation Loss: We define a domain adaptation loss that measures the distance of the feature space between the source and target domains. This loss is based on MMD [27] as used in [29], [30], [31]. MMD projects the feature of each data point to a Reproducing Kernel Hilbert Space (RKHS) to construct a mean embedding. Then the distance is computed between the constructed mean embeddings. The expression for the source feature representation is denoted by $\mathcal{F}_i^s = F_\Theta(x_i^s)$ (where $i \in \{1, 2, ..., N_s\}$). Similarly, the expression for target stream is represented by $\mathcal{F}_j^t = \Psi_\Theta(F_\Theta(x_j^t))$ (where $j \in \{1, 2, ..., N_t\}$). The calculation of MMD between the feature representations can be expressed as:

$$\mathcal{L}_{mmd}(\Theta) = \left\| \sum_{i=1}^{N_s} \frac{\chi(\mathcal{F}_i^s)}{N_s} - \sum_{j=1}^{N_t} \frac{\chi(\mathcal{F}_j^t)}{N_t} \right\|^2 \quad (4)$$

where $\chi(\cdot)$ denotes the mapping to RKHS. Eq. 4 can be further expanded to replace the inner products by kernel values based on the kernel trick and is expressed as:

$$\mathcal{L}_{mmd}(\Theta) = \Delta(\mathcal{F}_i^s, \mathcal{F}_{i'}^s, N_s, N_s) - 2 * \Delta(\mathcal{F}_i^s, \mathcal{F}_j^t, N_s, N_t) + \Delta(\mathcal{F}_j^t, \mathcal{F}_{j'}^t, N_t, N_t), \text{where} \quad (5)$$

$$\Delta(\mathcal{F}_i^s, \mathcal{F}_{i'}^s, N_s, N_s) = \sum_{i,i'} \frac{k(\mathcal{F}_i^s, \mathcal{F}_{i'}^s)}{(N_s)^2} \quad (6)$$

$$\Delta(\mathcal{F}_i^s, \mathcal{F}_j^t, N_s, N_t) = \sum_{i,j} \frac{k(\mathcal{F}_i^s, \mathcal{F}_j^t)}{(N_s N_t)} \quad (7)$$

$$\Delta(\mathcal{F}_j^t, \mathcal{F}_{j'}^t, N_t, N_t) = \sum_{j,j'} \frac{k(\mathcal{F}_j^t, \mathcal{F}_{j'}^t)}{(N_t)^2} \quad (8)$$

Here $k(\cdot, \cdot)$ is a kernel function defined as $k(x, x') = \exp(-\|x - x'\|^2/\sigma)$, with bandwidth parameter σ. In our work, we fix σ to be 1.

For the unsupervised domain adaptation, we ignore the loss term $\mathcal{L}_{target}(\Theta)$ since we do not have labeled images in the target domain. So the loss function in UDA can be written as:

$$\mathcal{L}_{UDA}(\Theta) = \mathcal{L}_{source}(\Theta) + \alpha\, \mathcal{L}_{mmd}(\Theta) \quad (9)$$

For the supervised case, the loss function is identical the same as Eq. 1. The only difference is that N_t in Eq. 4 will be K in this case.

Table I
DIFFERENT DOMAIN ADAPTATION SETTINGS IN OUR EXPERIMENT. FOR EACH SETTING, WE SHOW THE NUMBER OF LABELED TRAINING IMAGES IN THE SOURCE DOMAIN (I.E., N_s), THE NUMBER OF IMAGES IN THE TARGET DOMAIN (N_t), AND THE NUMBER OF TEST IMAGES IN THE TARGET DOMAIN. FOR SDA AND SSDA IN EACH SETTING, THE NUMBER OF LABELED TRAINING IMAGES IN THE TARGET DOMAIN (I.E., K) IS $\{1, 5, 10\}$.

Source→Target	$\mathbf{N_s}$	$\mathbf{N_t}$	**No. of test images**
ShanghaiTechA→Mall	300	800	1200
Mall→ShanghaiTechA	100	300	182
UCSD→Mall	100	800	1200
Mall→UCSD	100	800	1200
UCSD→TRANCOS	100	403	421
Mall→TRANCOS	100	403	421

Table II
AN ILLUSTRATION OF THE DOMAIN SHIFT PROBLEM IN COMMONLY USED COUNTING DATASETS. WHEN LEARNING THE MODEL ON THE TRAINING IMAGES OF A DATASET AND TESTING THE MODEL ON THE TEST DATA OF THE SAME DATASET, THE PERFORMANCE IS GENERALLY GOOD (FIRST FOUR ROWS). HOWEVER, WHEN WE TAKE A MODEL LEARNED FROM THE TRAINING IMAGES OF ONE DATASET AND TEST ON THE TEST IMAGES IN A DIFFERENT DATASET, THE PERFORMANCE DROPS DRAMATICALLY (LAST SIX ROWS).

Train	**Test**	**MAE**	**MSE**
ShanghaiTechA	ShanghaiTechA	110	169
UCSD	UCSD	1.07	1.35
Mall	Mall	2.60	3.32
Trancos	Trancos	5.53	7.39
Mall	ShanghaiTechA	407	683
ShanghaiTechA	Mall	22.67	23.24
UCSD	Mall	24.25	25.17
Mall	UCSD	11.26	11.55
UCSD	TRANCOS	22.08	30.52
Mall	TRANCOS	21.15	26.59

V. EXPERIMENTS

In this section, we first give details of the datasets and experiment setup in Sec. V-A. In Sec. V-B, we define several baseline methods for comparison, which are followed by our experimental results in Sec. V-C. Also, to demonstrate our method, we perform a cross-object domain adaptation experiment. In this setting, the source and target domains involve counting objects of different classes. Specifically, we consider a source domain involving people counting, while the target domain is the task of counting cars. The motivation of this cross object domain adaptation is that current crowd counting datasets are limited to a handful of object categories (e.g., mostly people [2], [3], [5]). However, in real-world applications, we might be interested in counting other objects in images. If we can perform well in cross-object domain adaptation, we can use the existing object counting datasets to help to count other objects in the target domain.

A. Datasets and Setup

Datasets: We use 4 datasets in the experiment, including ShanghaiTech Part_A(SHA) [10], Mall [3], UCSD [2] and TRANCOS [4]. ShanghaiTechA is a dataset containing dense crowds of people, while Mall and UCSD contain relatively sparse crowds. TRANCOS is a dataset of vehicle counting. We use these datasets to evaluate our approach when adapting between domains of different cameras (with different viewpoints, illumination conditions, etc.), density levels, and even between objects. For all datasets, we use the standard training/testing splits. From the training splits, we then perform domain adaptation by using different subsets of these datasets as shown in Table I. For example, the setup "UCSD→Mall" corresponds to using the UCSD dataset as the source domain and the Mall dataset as the target domain. For simplicity, we randomly select 100 labeled images ($N_s = 100$) as the source domain. The last column in Table I shows the total number of test images in the target domain. For the supervised (SDA) and semi-supervised (SSDA) domain adaptation, we consider the number of labeled images (i.e., K) to be $\{1, 5, 10\}$.

Generation of Ground-truth Density Maps: Similar to [10], we generate the ground-truth density maps for supervised learning from the provided point annotations (i.e., the center of the head of a person) in each of the datasets. The process involves applying a Gaussian kernel to the point annotations and normalizing the result to have a sum of one.

Evaluation Metrics: We use Mean Squared Error (MSE) and Mean Absolute Error (MAE) as the evaluation metrics. They are defined as follows:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{g}_i - g_i|, \quad \text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{g}_i - g_i)^2 \quad (10)$$

where $\hat{g}_i$ and g_i denote the predicted and ground-truth counts of the i-th test image, respectively. The predicted count $\hat{g}_i$ is obtained by integrating over the predicted density map.

Implementation Details: To speed up the training, we first pre-train the model only using the training images in the source domain by minimizing $\mathcal{L}_{source}(\Theta)$ defined in Eq. 2. We then fix the model parameters in the backbone network except for the fusion and the 1×1 conv layer. The MMD loss is defined on the output of the 1×1 conv layer. We use a learning rate of $1e-5$, a momentum of 0.9, a batch size of 1 and the Adam optimizer. Since the original MCNN code from [10] is not publicly available, we use a 3rd-party implementation of MCNN [1], which gives similar performance to that reported in [10] (see Table II).

[1]https://github.com/svishwa/crowdcount-mcnn

Table III

QUANTITATIVE RESULTS FOR THE SUPERVISED (SDA) AND SEMI-SUPERVISED (SSDA) DOMAIN ADAPTATION SETTINGS. WE COMPARE OUR MODELS ("OURS(SDA)" AND "OURS(SSDA)") AGAINST THE TWO BASELINE MODELS REPRESENTED AS "PRE-TRAINED" AND "FINE-TUNING". WE COMPARE THE RESULTS USING MAE AND MSE METRICS FOR K-SHOTS, WHERE $K \in \{1, 5, 10\}$. LOWER MAE/MSE MEANS BETTER PERFORMANCE. THE FIRST TWO ROWS CORRESPOND TO DOMAIN ADAPTATION BETWEEN SHANGHAITECHA (DENSE CROWD) AND MALL (SPARSE CROWD) DATASETS WHERE THE SOURCE AND TARGET DOMAINS HAVE DIFFERENT DENSITY LEVELS. THE 3RD AND 4TH ROWS SHOW THE ADAPTATION BETWEEN MALL (SPARSE CROWD) AND UCSD (SPARSE CROWD) WHERE TWO DOMAINS HAVE SIMILAR DENSITY LEVELS, BUT THE CAMERAS CAPTURE DIFFERENT ENVIRONMENTS WITH DIFFERING VIEWPOINTS, ETC. THE LAST TWO ROWS CORRESPOND TO CROSS-OBJECT DOMAIN ADAPTION, WHERE THE SOURCE DOMAIN (UCSD, MALL) IS FOR PERSON COUNTING AND THE TARGET DOMAIN (TRANCOS) IS FOR VEHICLE COUNTING. THE 1ST AND 2ND BEST RESULTS IN EACH SETTING ARE HIGHLIGHTED IN RED AND BLUE, RESPECTIVELY.

Source→Target	**Models**	**1-shot** ($K = 1$)		**5-shot** ($K = 5$)		**10-shot** ($K = 10$)	
		MAE	MSE	MAE	MSE	MAE	MSE
ShanghaiTechA → Mall	pre-trained	22.67	23.24	22.67	23.24	22.67	23.24
	fine-tuning	5.61	7.04	5.58	6.98	5.56	6.92
	Ours (SDA)	**5.36**	**6.65**	**5.34**	**6.62**	**5.42**	**6.72**
	Ours (SSDA)	**5.30**	**6.53**	**5.32**	**6.59**	**4.98**	**6.49**
Mall → ShanghaiTechA	pre-trained	407	683	407	683	407	683
	fine-tuning	392.98	661.98	390.22	659.10	392	658.166
	Ours (SDA)	**388.21**	**639.54**	**388.60**	**642.72**	**386.22**	**640.30**
	Ours (SSDA)	**384.83**	**641.94**	**382.56**	**639.50**	**382**	**639.88**
UCSD → Mall	pre-trained	24.25	25.17	24.25	25.17	24.25	25.17
	fine-tuning	3.45	4.34	3.40	4.30	3.42	4.33
	Ours (SDA)	**2.95**	**3.65**	**3.07**	**3.77**	**2.95**	**3.63**
	Ours (SSDA)	**2.93**	**3.62**	**2.92**	**3.60**	**2.92**	**3.64**
Mall → UCSD	pre-trained	11.26	11.55	11.26	11.55	11.26	11.55
	fine-tuning	2.43	3.70	2.45	3.72	2.44	3.80
	Ours (SDA)	**2.38**	**3.49**	**2.39**	**3.50**	**2.38**	**3.56**
	Ours (SSDA)	**2.36**	**3.57**	**2.35**	**3.48**	**2.35**	**3.55**
UCSD → TRANCOS	pre-trained	22.08	30.52	22.08	30.52	22.08	30.52
	fine-tuning	19.21	24.64	19.22	24.65	19.22	24.61
	Ours (SDA)	**17.85**	**23.14**	**17.07**	**22.49**	**17.06**	**22.34**
	Ours (SSDA)	**17.07**	**22.10**	**17.04**	**22.07**	**17.03**	**22.06**
Mall → TRANCOS	pre-trained	21.15	26.59	21.15	26.59	21.15	26.59
	fine-tuning	17.58	**22.97**	17.59	22.99	17.58	22.97
	Ours (SDA)	**17.54**	23.04	**17.55**	**22.92**	**17.58**	**22.92**
	Ours (SSDA)	**17.49**	**22.72**	**17.48**	**22.69**	**17.46**	**22.68**

B. Baselines

There is no previous work with which we can directly compare our method. Nevertheless, we define several baselines by ourselves, so that we can compare and demonstrate the effectiveness of the proposed approach. These baselines are the default methods that might currently be used in practice.

Pre-trained: The baseline method ignores the data (either labeled or unlabeled) in the target domain. It uses the labeled source training images to learn the model parameters. Then it directly applies the pre-trained model to the test data in the target domain.

Fine-tuning: This baseline is the standard fine-tuning technique commonly used in deep learning. First, we train the model using the labeled images in the source domain to generate a pre-trained model. Then we fine-tune the model on the small number of labeled images in the target domain. Note that this baseline is only applicable in the SDA and SSDA settings since these two settings assume the target domain has some labeled training images. However, it does not work in the UDA setting in which there are no labeled training images in the target domain.

C. Experimental Results

Before presenting our main experimental results, we first perform an experiment as a sanity check to illustrate the domain shift problem in commonly used crowd counting datasets. This will demonstrate the need for domain adaptation in crowd counting. In Table II, we show the quantitative results using training/test datasets from different combinations of various datasets. We see that the performance is generally good when both training and test data are from the same dataset. However, when we train the model using one dataset and test on another dataset, there is a significant drop (on the order of 10-20 times) in performance. This shows that there is a notable domain shift between the common datasets used in crowd counting. Without explicit domain

adaptation, a model trained on one dataset (i.e. source) usually does not generalize well to another different dataset (i.e. target).

Table III shows the results of supervised and semi-supervised domain adaptation when $K = 1, 5, 10$. We compare our approach ("Ours (SDA)" and "Ours (SSDA)") with the two baselines ("pre-trained" and "fine-tuning"). Our proposed method significantly outperforms the baselines. In most cases, our semi-supervised method performs the best. This is reasonable since this setting makes the best use of all the training information available. Interestingly, there is no significant difference among the results when we vary the value of K from 1 to 10. One possible explanation is that the value of K is relatively small. When we increase K from 1 to 10, it is not enough to have a significant impact on the learning.

We present the results of unsupervised domain adaptation in Table IV. We compare our method with the baseline which directly applies the pre-trained model from the source domain to the target domain. Our method significantly outperforms this baseline. Note that we cannot compare with the "fine-tuning" baseline as it requires access to labeled training images in the target domain.

Table IV
QUANTITATIVE RESULTS OF THE UNSUPERVISED DOMAIN ADAPTATION (UDA), WHERE THE TARGET DOMAIN ONLY HAVE UNLABELED TRAINING IMAGES. THE BEST RESULTS ARE HIGHLIGHTED IN RED.

Source→Target	Models	MAE	MSE
ShanghaiTechA → Mall	pre-trained	22.67	23.24
	Ours (UDA)	**5.61**	**7.04**
Mall → ShanghaiTechA	pre-trained	407.00	683.00
	Ours (UDA)	**392.98**	**661.88**
UCSD → Mall	pre-trained	24.25	25.17
	Ours (UDA)	**2.93**	**3.65**
Mall → UCSD	pre-trained	11.26	11.55
	Ours (UDA)	**2.52**	**3.38**
UCSD → TRANCOS	pre-trained	22.08	30.52
	Ours (UDA)	**17.19**	**22.14**
Mall → TRANCOS	pre-trained	21.15	26.59
	Ours (UDA)	**17.90**	**23.36**

D. Analysis

We perform additional analysis to get further insight into our proposed approach. Considering two oracle methods to be the upper bound of our approach. The difference between these two oracle methods is as follows. Oracle-1 only considers the training data in the target domain and it can overfit to this domain. Oracle-2 is trained with more training data since it uses training images from source and target domains. However, Oracle-2 is not explicitly trained to fit the target domain.

In Table V, we compare our domain adaptation approach to these oracle methods. Note that in the case of SDA and SSDA, we have used the result from the 1-shot ($K = 1$) setting. We can make several interesting observations. First, Oracle-1 consistently outperforms Oracle-2 on all datasets. This suggests that if we have enough labeled data in the target domain and if we only care about the model performance in the target domain, we should directly use standard supervised method trained in the target domain. The benefits of domain adaptation are more relevant when the labelled data in the target domain is scarce. Second, the difference between Oracle-2 and our approach is not significant. This means if we use the proposed domain adaptation method, the number of labeled images in the target domain does not significantly impact the performance of the model. This is in fact good news for practical applications, since it implies that annotating a small number of images in the target domain is enough to achieve competitive performance.

Table V
COMPARISON OF OUR DOMAIN ADAPTATION APPROACHES WITH TWO ORACLE METHODS. THE 1ST AND 2ND BEST RESULTS IN EACH SETTING ARE HIGHLIGHTED IN RED AND BLUE, RESPECTIVELY.

Target	Method	Source	MAE	MSE
SHA	Oracle-1	-	**110**	**169**
	Oracle-2	Mall	**369.51**	**486.66**
	Ours (UDA)	Mall	392.98	661.88
	Ours (SDA)	Mall	388.21	639.54
	Ours (SSDA)	Mall	384.83	641.94
Mall	Oracle-1	-	**2.60**	**3.32**
	Oracle-2	SHA	**5.23**	**6.42**
	Ours (UDA)	SHA	5.61	7.04
	Ours (SDA)	SHA	5.36	6.65
	Ours (SSDA)	SHA	5.30	6.53
Mall	Oracle-1	-	**2.60**	**3.32**
	Oracle-2	UCSD	**2.90**	**3.60**
	Ours (UDA)	UCSD	2.93	3.65
	Ours (SDA)	UCSD	2.95	3.65
	Ours (SSDA)	UCSD	2.93	3.62
UCSD	Oracle-1	-	**1.07**	**1.35**
	Oracle-2	Mall	**2.28**	**3.37**
	Ours (UDA)	Mall	2.52	3.38
	Ours (SDA)	Mall	2.38	3.49
	Ours (SSDA)	Mall	2.36	3.57
TRANCOS	Oracle-1	-	**4.67**	**6.61**
	Oracle-2	UCSD	**16.98**	**21.05**
	Ours (UDA)	UCSD	17.19	22.14
	Ours (SDA)	UCSD	17.85	23.14
	Ours (SSDA)	UCSD	17.07	22.10
	Oracle-2	Mall	**17.43**	**22.65**
	Ours (UDA)	Mall	17.90	23.36
	Ours (SDA)	Mall	17.54	23.04
	Ours (SSDA)	Mall	17.49	22.72

VI. CONCLUSION

We have proposed deep learning-based domain adaptation in crowd counting. Our method can operate in supervised, unsupervised, and semi-supervised settings. The proposed approach has been demonstrated to outperform other alter-

native approaches experimentally on several public datasets. In addition to standard domain adaptation where the object of interest is the same in both source and target domains, we have shown that our method is also applicable for cross-object domain adaptation. Although we have used MCNN as the backbone network and MMD as the domain adaptation loss in this paper, we are certainly not limited to these specific choices – our model can be used with other backbone architectures and domain adaptation losses.

REFERENCES

[1] D. B. Sam, S. Surya, and R. V. Babu, "Switching convolutional neural networks for crowd counting," in *CVPR*, 2017.

[2] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *CVPR*, 2008.

[3] K. Chen, C. C. Loy, S. Gong, and T. Xiang, "Feature mining for localised crowd counting." in *BMVC*, 2012.

[4] R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. López-Sastre, S. Maldonado-Bascón, and D. Onoro-Rubio, "Extremely overlapping vehicle counting," in *Iberian Conference on Pattern Recognition and Image Analysis*, 2015.

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

[6] C. C. Loy, S. Gong, and T. Xiang, "From semi-supervised to transfer counting of crowds," in *ICCV*, 2013.

[7] M. Marsden, K. McGuinness, S. Little, C. E. Keogh, and N. E. O'Connor, "People, penguins and petri dishes: Adapting object counting models to new visual domains and object types without forgetting," in *CVPR*, 2018.

[8] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE PAMI*, 2012.

[9] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *IJCV*, 2013.

[10] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *CVPR*, 2016.

[11] M. Hossain, M. Hosseinzadeh, O. Chanda, and Y. Wang, "Crowd counting using scale-aware attention networks," in *WACV*, 2019.

[12] V. A. Sindagi and V. M. Patel, "Generating high-quality crowd density maps using contextual pyramid CNNs," in *ICCV*, 2017.

[13] X. Liu, J. van de Weijer, and A. D. Bagdanov, "Leveraging unlabeled data for crowd counting by learning to rank," in *CVPR*, 2018.

[14] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE PAMI*, 2006.

[15] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML*, 2015.

[16] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, 2015.

[17] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," in *NeurIPS*, 2016.

[18] M. A. Hossain, M. Kumar, M. Hosseinzadeh, O. Chanda, and Y. Wang, "One-shot scene-specific crowd counting," in *BMVC*, 2019.

[19] G. Csurka, "Domain adaptation in computer vision applications: A comprehensive survey," *arXiv preprint arXiv:1702.05374*, 2017.

[20] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, 2018.

[21] L. Bruzzone and M. Marconcini, "Domain adaptation problems: A dasvm classification technique and a circular validation strategy," *IEEE PAMI*, 2010.

[22] W.-S. Chu, F. De la Torre, and J. F. Cohn, "Selective transfer machine for personalized facial action unit detection," in *CVPR*, 2013.

[23] M. Gheisari and M. S. Baghshah, "Unsupervised domain adaptation via representation learning and adaptive classifier learning," *Neurocomputing*, 2015.

[24] B. Gong, K. Grauman, and F. Sha, "Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation," in *ICML*, 2013.

[25] S. Pachori, A. Deshpande, and S. Raman, "Hashing in the zero shot framework via domain adaptation," *Neurocomputing*, 2018.

[26] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, 2011.

[27] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *NeurIPS*, 2007.

[28] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, "Correcting sample selection bias by unlabeled data," in *NeurIPS*, 2007.

[29] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *ICML*, 2015.

[30] A. Rozantsev, M. Salzmann, and P. Fua, "Beyond sharing weights for deep domain adaptation," *IEEE PAMI*, 2018.

[31] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv:1412.3474*, 2014.