Latent Domain Generation for Unsupervised Domain Adaptation Object Counting

Anran Zhang, Yandan Yang, Jun Xu, *Member, IEEE*, Xianbin Cao, Xiantong Zhen, and Ling Shao, *Fellow, IEEE*

***Abstract*—Unsupervised cross-domain object counting has recently received great attention in computer vision, which generalizes the model from the source domain to the unlabeled target domain. However, it is an extremely challenging task because only unlabeled data is available from the target domain and the domain gap between two domains is implicit in object counting. In this paper, we propose a latent domain generation method to improve the generalization ability of unsupervised domain adaptation object counting by generating a latent domain. To this end, we propose a domain generator with random perturbations to learn a new latent distribution derived from the original source distribution. The latent domain generator can extract target information sampled in its stochastic latent representation, which preserves the original target information and enhances the diverse ability. Meanwhile, to ensure that the generated latent domain is consistent with the source domain in counting performance, we introduce a consistency loss to encourage similar output from latent and source domains. Moreover, to enhance the adaptation ability of the generated latent domain, we apply the adversarial loss to achieve alignment between the latent and target domains. The domain generator with the adversarial loss and consistency loss ensures that the generated domain is aligned to the target while also improving the robustness of the original source domain model. The experiment indicates that our framework can effortlessly extend to scenarios with different objects (crowd, cars). The experiments also demonstrate the effectiveness of our method on unsupervised realistic-to-realistic crowd counting problems.**

***Index Terms*—Object counting, domain adaptation, unsupervised learning.**

I. Introduction

OBJECT counting is to estimate the number of objects in a region of scene, and is important in computer vision fields [5], [24], [30], [32], [52]. The populations of statistical scenes and their density distributions are invaluable for public management, health monitoring, and security analysis [45], [46]. For example, monitoring the number of objects is a security problem for the public in dense scenarios, such as bus stations and shopping malls. Existing counting methods can be roughly divided into three categories: detection based methods [10], [55], regression based methods [20], [27], and density map estimation based methods [2], [68]. However, these methods [4], [6], [27], [68] may not generalize well to unseen scenarios, especially when there is a domain gap between the training (source) and test (target) images. In such cases, labeling per-pixel annotations for an unseen target dataset would entail prohibitively high labor costs for re-training a new model.

To address the issues above, several unsupervised domain adaptation (UDA) [12], [19], [69] is widely studied to alleviate the dependency on the labeling and domain shift problem. UDA is able to learn cross-domain generalization, and reduce the burden on labeling the target domain. In object counting, UDA is mainly implemented under the adversarial learning framework [29], [59]. CODA [29] directly aligns the same distribution between the source and target domain to adapt the model across different object categories datasets. SE-CycleGAN [59] can solve the explicit domain gap (style) in synthesis-to-real crowd scenes, transforms source data into a seen target scene. However, there are complex and implicit gaps between different counting datasets, such as noisy scenes, various scales, and multi-level densities, especially for different object categories (people, cars). Therefore, it is essential to generate a new domain from the source and target domain, which can be adaptive to the counting model.

To tackle these limitations, in this paper, we propose a latent domain generation method to learn a new domain for unsupervised domain adaptation object counting. We generate a new domain from the source and target domain, as perturbations to the original source domain, to improve the generalization ability of the pretrained model. Motivated by the unsupervised loss in semi-supervised learning [48], which penalized different predictions for the same training input with stochastic perturbations, to improve the generalization ability for the model trained with the limited label. Unlike the unsupervised loss [48] in which the perturbation is stochastic, the proposed new domain has the direction that is near the target domain and contains stochastic characteristics. To generate a new domain, we propose a *domain generator* with the objective of learning a new domain,

Manuscript received 16 August 2021; revised 6 February 2022; accepted 16 March 2022. Date of publication 28 March 2022; date of current version 7 June 2023. This work was supported in part by the National Key Scientific Instrument and Equipment Development Project under Grant 61827901, in part by the Fundamental Research Funds for the Central Universities of Nankai University 63211099, and in part by the Natural Science Foundation of China under Grants 91738301, 62002176, and 62088101. The Guest Editor coordinating the review of this manuscript and approving it for publication was Mr. Chuang Gan. *(Corresponding author: Xianbin Cao.)*

Anran Zhang, Yandan Yang, and Xianbin Cao are with the Key Laboratory of Advanced Technology of Near Space Information System, Beihang University, Beijing 100191, China (e-mail: zhanganran@buaa.edu.cn; yangyandan@buaa.edu.cn; xbcao@buaa.edu.cn).

Jun Xu is with the School of Statistics and Data Science, Nankai University, Tianjin 300071, China (e-mail: nankaimathxujun@gmail.com).

Xiantong Zhen is with the AIM Lab, University of Amsterdam, 1012 WX Amsterdam, The Netherlands (e-mail: zhenxt@gmail.com).

Ling Shao is with the Inception Institute of Artificial Intelligence, Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE (e-mail: ling.shao@ieee.org).

Digital Object Identifier 10.1109/TMM.2022.3162710

derived from the source and target domain. Our domain generator combines an auto-encoder with a random sampling method for the target domain, which delivers generalization ability in cross-domain counting datasets. The stochastic target characteristics are encoded in a low-dimensional latent space, combining the source information to generate a new domain derived from the original two domains. To make the counting model robust to the generated domain, we apply the *consistency loss* between the generated and original source domain outputs space to ensure consistency in these two domains. We further apply an *adversarial loss* in the feature space to align the distribution between the generated domain and the target domain. With the constraint of the consistency loss and the adversarial loss, the generated domain is to simultaneously approach the source and target domain, and to further make the unlabeled target domain adaptive to the labeled source counting model.

The contributions are summarized as follows:

- We propose a novel domain generation method, which can improve the generalization ability of unsupervised crowd counting by learning new latent domain distributions between the source and target domain in a novel *domain generator*.
- We propose to employ the *consistency loss* and *adversarial loss* for the *domain generator* to ensure that the generated domain is close to the source and target ones.
- Extensive experiments show that our method has a significant improvement in unsupervised domain adaptation object and crowd counting. And the ablation study demonstrates the effectiveness of our proposed approach.

The remaining of this paper is organized as follows. We introduces related works in Section II. The proposed Latent Domain Generation (LDG) method is described in Section III. We reports the experimental results and ablation study in Section IV. The conclusion is presented in Section V.

II. Related Work

A. (Semi-)Supervised Crowd Counting

Supervised Crowd Counting counts the number of people. It relies on labeling the location of each person in the region and depends on the detection, regression, and density-based methods. Early works [4], [6], [10], [27], [55] generally use a method based on detection and regression to count the total number of people in the scene. These methods are typically hindered by the intrinsic challenges of the occlusions, undersize, low-quality issues, and the holistic features [10], [55] of individuals in a scene. Recent detection based methods [10], [22], [37], [38], [55] had manually annotated the bounding boxes or point supervision on the dataset and trained a detection network for crowd counting. Current density map based methods [1], [8], [35], [50], [57], [63], [68] have shown great success in crowd counting, which first generate density map ground-truth and then assigned crowd counting as a pixel regression problem. In density-based methods, some approaches are aimed at exploiting multi-scale features or multi-context information to deal with the people scale variation problem [2], [3], [31], [68]. Zhang *et al.* [68] and Sam *et al.* [47] employed multi-column networks with various kernel sizes to gain different receptive fields for each people. To explore the structural information in the density map, SANet [3] tried to measure the structural similarity between ground-truth and output density maps by exploring each local information during computing loss. With the success of attention mechanisms, RAZN [31] proposed to iteratively locate regions with high ambiguity in high-resolution space. These methods count the number by the density map that can also provide the location information of the crowd, which are suitable for both sparse and crowded scenes.

Semi-supervised Crowd Counting is the method to learn with a small amount of labeled data and a large amount of unlabeled data for crowd counting [29], [44], [59]. Tan *et al.* [52] extracted sequential information between unlabelled samples and their temporally neighboring samples to build a semi-supervised regularization. To further explore intrinsic structures of unlabelled data, Loy *et al.* [5] proposed an active and semi-supervised regression model, by the manifold regularization to assimilate the count estimation of two nearby crowd pattern points in the manifold. These methods can learn similar visual characteristics among the most informative frames, which show the transferability performance in sparse and simple video counting scenes. With recent demands on counting dense scenes, few-shot counting methods resort to use a few labeled data in a query set when transferring the model from the support set. Wang *et al.* [61] exploited few labeled target data to learn two matrices (product factor and bias) for each source neuron via a linear transformation in the parameter level. Zhao *et al.* [51] iteratively annotated the most informative images in an active learning framework to learn the counting model over unlabeled data.

B. Unsupervised Crowd Counting

Recently, due to the demand for crowd counting in large-scale dense crowds, the labeling burden becomes more onerous than other visual tasks in large-scale dense crowds. Some methods address the labeled burden on specific datasets, such as unsupervised domain adaptation (UDA) [12], [13], [19], [29], [59], [60], [69]. These methods aim to improve the performance in the unlabeled target domain by using source data. UDA [19], [53], [56], [69] can only utilize the target unlabeled data to eliminate the labeled burden. The general idea of UDA usually learns the domain-invariant features between source and target domain by minimizing the difference of the distributions. Recent unsupervised crowd counting works [29], [59] utilized the pre-trained model on labeled source domains to generalize a model for unlabeled target domains based on adversarial learning. Wang *et al.* [59] proposed the SE CycleGAN method, which translates synthetic data to photo-realistic crowded scenes. CODA [29] addressed the UDA problem in object counting via scale aware adversarial learning to align scene layout and local context between source and target (crowd, cars) images.

Although these GAN and CycleGAN methods [29], [59] can solve the explicit domain gap (style) in synthesis-to-real, there still exist major problems, such as the imitations of transferring styles and the limited synthetic scenes. As the style is not the only problem of domain gaps, there are also other implicit gaps

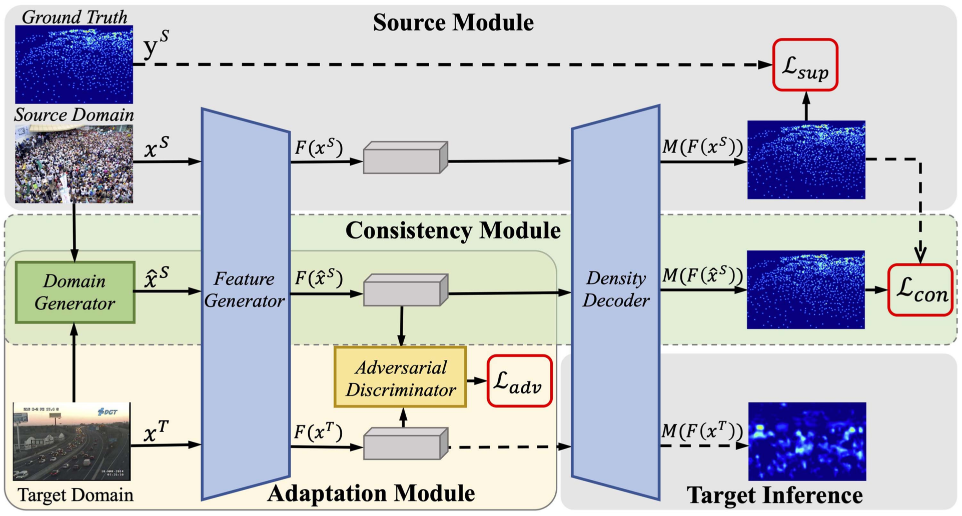

Fig. 1. Overview of our proposed Latent Domain Generation (LDG) Method. The overall framework includes four blocks: Feature Generator, Domain Generator, Adversarial Discriminator, and Density Decoder. Feature Generator extracts the base feature for counting; Domain Generator use the auto-encoder to obtain a generated domain; Adversarial Discriminator learn the domain-invariation between the generated domain and target domain; Density Decoder decodes the final density maps. To empower these blocks, there are three training losses in our method: **Supervised Loss ($\mathcal{L}_{\mathbf{sup}}$), Consistency Loss ($\mathcal{L}_{\mathbf{con}}$) and Adversarial Loss** ($\mathcal{L}_{\mathbf{adv}}$). $\mathcal{L}_{\mathbf{sup}}$ ensures that the feature generator and density decoder for counting is accurate; $\mathcal{L}_{\mathbf{con}}$ ensures that the model is robust to data in both source domain and generated domain; $\mathcal{L}_{\mathbf{adv}}$ encourages the generated domain to align to the target domain.

(noisy scenes, various scales, and multi-level densities, different types of objects). And the synthetic dataset proposed in [59], with only 100 scenes except for similar places, which is limited and simple compared with the real dataset (eg., UCF-QNRF [21] has probably 1535 diverse scenes). In the problem of realistic-to-realistic adaptation, the scenes are drastically changing, and the objects are also diverse. Our LDG first can learn a latent domain derived from the source and target domain in an auto-encoder, and can also increase the diversity of source domains due to its stochastic latent representation in the sampling process of the auto-encoder. Experiment results demonstrate that LDG has significant effects in realistic-to-realistic adaptation problems, additionally for the different objects counting problem.

C. Domain Adaptation

Domain Adaptation is a well researched topic for the task of classification, detection and semantic segmentation [11], [39], [43], [58], [62], [64]. Domain Adaptation aims to maximize the performance on a given target domain using labeled source domains [58]. Recent works put their efforts to bridge the gap between the domains mostly based on adversarial learning methods [29], [53], [56] or style-transfer solution [59] in object counting and semantic segmentation. Some works [14]–[16] have shown that generating intermediate domains is helpful to bridge the distribution gap between source and target domains. Gong *et al.* [15] proposed a domain flow generation (DLOW) model to improve the generalization ability of learned models, by generating a continuous sequence of intermediate domains flowing shifting from the source domain to the target domain. Yang *et al.* [65] proposed a Bi-Directional Generation domain adaptation model to synthesize the intermediate domains conditioned on each domain, in which the augmented samples play as a bridge to reduce the domain discrepancy. Compared to the domain generation works, our work aims to use a domain generator with random perturbations to generate a latent domain, to leverage the task of domain adaptation network as its own guide toward useful generated domains that enhances the generalization capability of the model.

III. Latent Domain Generation (LDG) Method

In this section, we first introduce the preliminary of unsupervised cross-domain crowd counting, which contains the overall structure of our proposed method and the representation of each component in Section III-A. We second propose the *Latent Domain Generator* in Section III-B. We next describe our proposed two methods, *Learning Consistency in Source Domains* in Section III-C, and *Learning Cross-Domain Similarity* in Section III-D. Finally we provide the implementation details. An overview of our proposed Latent Domain Generation (LDG) method is illustrated in Fig. 1.

A. Preliminary

Task formulation: Given a labeled dataset in a source domain and an unlabeled dataset in a target domain, the goal of unsupervised cross-domain crowd counting is to train a counting model from the source domain that generalizes well on the target domain. However, the domain shift problem (e.g., scenes, crowd levels, and scale variations) makes the pretrained model on the source domain overfit on the target domain in previous works [29], [59]. Direct aligning the source and the target domain is difficult and ambiguous. Therefore, our goal is to find the latent domain from the source and target domain to help adaptation.

Let $X_S = \{x_i^S, y_i^S\}$ represents a labeled dataset from the source domain, where $i = 1, 2, \ldots, K$. K is the number of samples in the source dataset. x_i^S and y_i^S are the source image sample

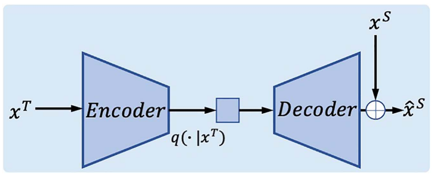

Fig. 2. $G(x;\theta_g)$ is our proposed latent domain generator. Each source domain sample x^S will be transformed into $\hat{x}^S$ with the target information, which represents the new generated domain image.

and ground truth density map, respectively. $X_T = \{x_i^T\}$ represents the unlabeled dataset, where $i = 1, 2, \ldots, M$, M is the number of samples in the target dataset. x_i^T is the target image sample without its ground truth density map.

$$\{X_S, X_T\} \Rightarrow T_{UDA}. \tag{1}$$

T_{UDA} represents the task of the unsupervised domain adaptation. The goal of T_{UDA} is to improve the generalization ability of the source model on the target domain without the labels of target domains.

Network components: Let $G(x;\theta_g)$ be a Domain Generator which is a function parameterized by θ_g which maps an image x to a generated representation $\hat{x}$. Let $F(x;\theta_f)$ be a feature generator parameterized by θ_f which maps an image x to a hidden representation h representing features that in a high-level dimension. Let $D(h;\theta_d)$ be a discriminator making the features from the two domains have a similar distribution. Finally, $M(h;\theta_m)$ represents a density map decoder function, parameterized by θ_m that maps from hidden representations h to the density map predictions y.

B. Latent Domain Generator

The auto-encoder [18] is used to reconstruct high-dimensional input vectors by low-dimensional codes, which is an efficient way to progressively reveal low-dimensional, nonlinear structure [9], [18], [26], [40], [54]. To simultaneously encode the target domain information and add randomness for the low-dimensional latent representation, we generate a latent domain that introduces randomness in the latent low-dimensional representation at extracting information of the target domain, with maintaining source information, which preserves the original target information and enhances the diverse ability. Let x^S and x^T be the image of objects from the source and target datasets. The domain generator is developed on an auto-encoder structure. $\hat{x}^S$ is the output of the domain generator $G(x;\theta_g)$, which is shown in Fig. 2. The formulation is defined as:

$$\hat{x}^S = G(x^T, x^S; \theta_g), \tag{2}$$

where each source domain sample x^S will be transformed into a disturbed sample $\hat{x}^S$ by the domain generator $G(x^T, x^S; \theta_g)$ with parameter θ_g.

To extract target information and increase the diversity of target information, we depatentvarioy the sampling operation, which is implemented as the reparameterization trick in the sampling process in $q(z|x^T)$. Here z is the latent variable, our method is to sample the stochastic latent representation in the target domain in a distribution $q(z|x^T)$, and to encode these information for the original source domain to obtain a new latent domain sample $\hat{x}^S$. The stochastic sampling method can enhance the diversity of extracted feature, while preserving the original target information, which can further enhance the diversity of the new generated domain. Since the generated latent source domain is expected to explore the target-related domain for the original source domain, we also adopt the adding operation to release the burden of domain generator which is shown in Fig. 2.

With our latent domain generator, the generated new domain can maintain the source information with the stochastic target domain characteristics. This is beneficial to the unsupervised domain adaptation. First, the newly generated domain can improve the robustness of the model by the **Supervised Loss** and **Consistency Loss** in the **Source Module** and **Consistency Module** in Fig. 1. In these processes, the stochastic sampling in extracting target characteristics can greatly improve the diversity of the newly generated domains, thereby improving the robustness of the model, while also preserving the original target characteristics. Second, in the **Adaptation Module**, it is more feasible to narrow the distance between the newly generated domain (the source information with the stochastic target domain characteristics) and the target domain, rather than directly narrowing the distribution of two different domains. Next, we introduce these in detail.

C. Learning Consistency in Source Domains

Π-model with the unsupervised loss in semi-supervised learning [48], is one of the popular approaches in semi-supervised learning, which encourages consistent network output between two realizations of the same input stimulus, under two different conditions. Π-model can make the model robust to the input noise, encourages consistent output for the input and the same input with noise, both for labeled data and unlabeled data. Inspired by this, we feed our original source domain input x^S and the generated domain input $\hat{x}^S$ to the supervised counting model, and we propose to employ the *consistency loss* to ensure the output of the generated domain is consistent with the original domain. In detail, the consistency loss computes pixel-wise mean square error loss between the final predicted density maps of both disturbed sample $\hat{x}^S$ and the source domain sample x^S:

$$\begin{aligned}\mathcal{L}_{con}^S = \frac{1}{N}\sum_{i=1}^{N} ||M(F(x^S;\theta_f);\theta_m))_i \\ - M(F(\hat{x}^S;\theta_f);\theta_m)_i||_2^2,\end{aligned} \tag{3}$$

where N denotes the total pixels, each sample x^S goes through a feature generator $F(x;\theta_f)$ and a density decoder $M(x,\theta_m)$. And the results on source domain sample x^S serves as a soft label for the disturbed sample $\hat{x}^S$.

D. Learning Cross-Domain Similarity

To align the generated latent domain with the target domain distribution, we propose an adaptation module with an adversarial loss to learn similarity between the generated domain and the target domain. The disturbed source image $\hat{x}^S$ for a source input image x^S is defined in (2) as the output of an auto-encoder $G(x^T, x^S; \theta_g)$. Our intuition is that the adversarial approach can narrow the gap between two domains, to make the generated and target domain share similarities in the inner representations.

Followed the unsupervised domain adaptation framework based on adversarial learning, the proposed model consists of three parts: (1) a *Domain Generator* $G(x^T, x^S; \theta_g)$ to encode target characteristics within the disturbed source images; (2) *Feature Generators* $F(\hat{x}^S; \theta_f)$ and $F(x^T; \theta_f)$ to share strong similarities in complex representations between the source and target domain; (3) an *Adversarial Discriminator* $D(h; \theta_d)$ to distinguish whether the input is from the latent source or target domain. Here, we introduce the training step for the adversarial loss: **Generator-step** for training generators and **Discriminator-step** for training the discriminator.

Generator-step: There are two generators in this step: the feature generator $F(x^S; \theta_f)$ and the domain generator $G(x^T, x^S; \theta_g)$ to ensure the consistency for the feature generator $F(x^S; \theta_f)$ and the density map decoder $M(F(x^S; \theta_f))$, which is implemented as consistency loss $\mathcal{L}^S_{con}$. The consistency loss $\mathcal{L}^S_{con}$ is crucial to ensure outputs consistency between the generator samples and the corresponding source images. Then adversarial loss $\mathcal{L}^T_{adv}$ fools the discriminator by maximizing the probability of the target feature being considered as the source feature. The overall loss is:

$$\begin{aligned}\mathcal{L}_{\mathcal{UDA}} = {} & \mathcal{L}^S_{sup}(M(F(x^S; \theta_f); \theta_m), y^S) \\ & + \mathcal{L}^S_{con}(M(F(x^S; \theta_f); \theta_m), M(F(\hat{x}^S; \theta_f); \theta_m)) \\ & + \mathcal{L}^T_{adv}(D(F(x^T; \theta_f); \theta_d), 1), \end{aligned} \tag{4}$$

where the $\hat{x}^S = G(x^T, x^S; \theta_g)$. $\mathcal{L}^S_{con}$ and $\mathcal{L}^S_{sup}$ are implemented as the ℓ_2 loss, that is the pixel-wise mean square error loss for pixel-level regression tasks. $\mathcal{L}^T_{adv}$ is implemented as the cross-entropy loss.

Discriminator-step: In this step, $D(\cdot; \theta_d)$ distinguishes whether the input is from the latent source domain or the target domain. $G(x^T, x^S; \theta_g)$ helps distinguish the target and the generated source features, motivated by the adversarial training strategy under the unsupervised domain adaptation framework [53]. The loss is defined as:

$$\begin{aligned}\mathcal{L}_{\mathcal{UDA}} = {} & \mathcal{L}^S_{adv}(D(F(\hat{x}^S; \theta_f); \theta_d), 1) \\ & + \mathcal{L}^T_{adv}(D(F(x^T; \theta_f); \theta_d), 0), \end{aligned} \tag{5}$$

where $\mathcal{L}^S_{adv}$ and $\mathcal{L}^T_{adv}$ represent the cross-entropy loss.

Motivated by unsupervised domain adaptation works [25], [29], [53], [56], we jointly train the counting networks and discriminators in one stage, with the ultimate goal of minimizing the counting loss in generators for source images, while learning consistency in the source domain and similarity in cross-domain. The overall algorithm is described in Algorithm 1.

Algorithm 1: LDG.

```
Require: $X_S = \{x^S_i, y^S_i\}$ source images and labels
Require: $G(\cdot; \theta_g)$ domain generator
Require: $F(\cdot; \theta_f)$ feature generator
Require: $M(\cdot; \theta_m)$ density decoder
Require: $D(\cdot; \theta_d)$ adversarial discriminator
Require: $X_T = \{x^T_i\}$ target domain images
 1: for each epoch do
 2:   for each minibatch do
 3:     if Step $Generator = True$ then
 4:       Compute loss followed the Generator-step as
          (4): $\mathcal{L}_{\mathcal{UDA}} \leftarrow$
          $\mathcal{L}^S_{sup}(x^S, y^S) + \mathcal{L}^S_{con}(x^S, \hat{x}^S) + \mathcal{L}^T_{adv}(x^T, 1)$
 5:     end if
 6:     if Step $Discrimintor = True$ then
 7:       Compute loss followed the Discriminator-step as
          (5): $\mathcal{L}_{\mathcal{UDA}} \leftarrow \mathcal{L}^S_{adv}(\hat{x}^S, 1) + \mathcal{L}^T_{adv}(x^T, 0)$
 8:     end if
 9:   end for
10: end for
```

E. Implementation Details

We use the general network (VGG-16 [49]) as the backbone for the feature generator. For the cross-domain counting in real-to-reals datasets, we first pre-train a model on a labeled source domain (e.g., ShanghaiTech Part A [68]), and save the pre-trained model to adapt to the unlabeled target domain.

The encoder of Domain Generator $G(x; \theta_g)$ first applies the kernel size 3 for two convolutional blocks which transform the channel from 3 to 64 to 256. Each convolutional block uses the Maxpool to downsample the feature map. Then we apply some residual convolutional blocks in channel 256. And the decoder does the operation to restore the image size corresponding to the input. The Domain Generator $G(x; \theta_g)$ network parameters are randomly initialized by a Xavier with a mean zero and a standard deviation of 0.01. And the Feature Generator $F(x; \theta_f)$ and the Density Decoder $M(h; \theta_m)$ network are initialized by the source pretrained model. Adam optimizer with a small learning rate of $1e-5$ is used to $G(x; \theta_g)$, $F(x; \theta_f)$ and $M(h; \theta_m)$, $1e-3$ is used to $D(h; \theta_d)$. Correspondingly, the hyper-parameter weight of $\mathcal{L}^S_{sup}$, $\mathcal{L}^S_{con}$ and $\mathcal{L}^T_{adv}$ are 1.0, 1.0 and 0.2. The network is trained with a batch size of 10 during pretraining on the source domain, and a batch size of 4 during adapting to the target domain. The implementation is based on Pytorch.

IV. Experiments

In this section, we conduct experiments on crowd-to-cars unsupervised object counting and real-to-real unsupervised crowd counting in Section IV-C1 and Section IV-C2, respectively. In ablation studies, we demonstrate the effectiveness of our proposed modules.

A. Experimental Setup

Data Preparation: Since annotations for crowd images are labeled at the center of the pedestrian head, we use the Gaussian kernel to convert these points to generate the crowd density map.

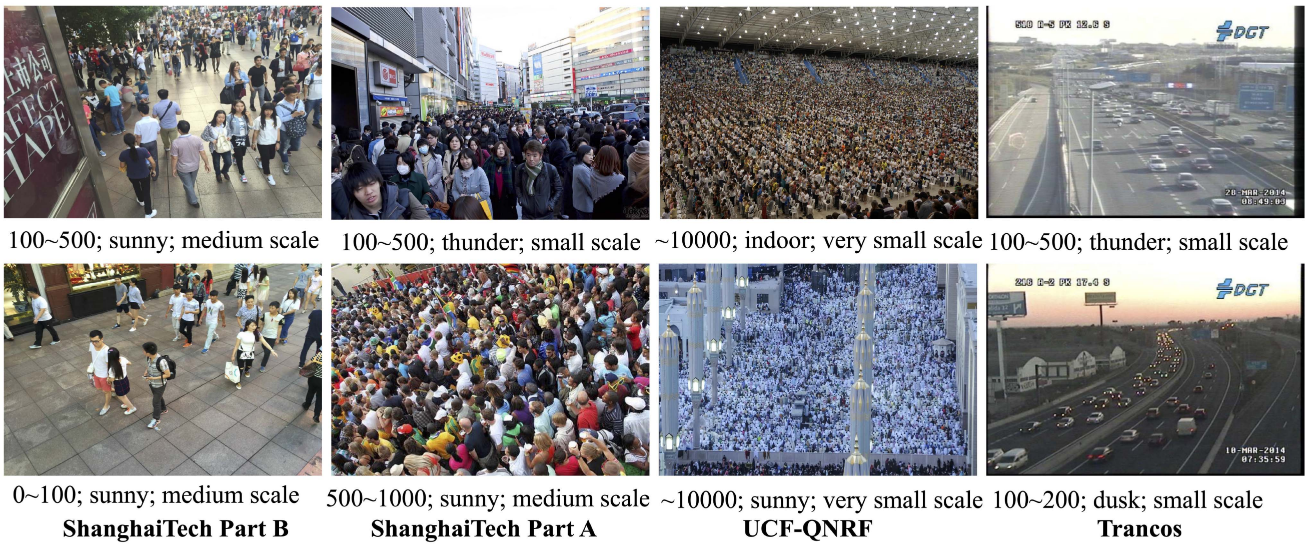
100~500; sunny; medium scale
100~500; thunder; small scale
~10000; indoor; very small scale
100~500; thunder; small scale
0~100; sunny; medium scale
500~1000; sunny; medium scale
~10000; sunny; very small scale
100~200; dusk; small scale
ShanghaiTech Part B
ShanghaiTech Part A
UCF-QNRF
Trancos

Fig. 3. Real-world crowd/car datasets. It can be seen that each dataset all has multiple densities, multiple scales, and complex background issues.

The normalized Gaussian kernel is defined as :

$$D(x) = \sum_{x_i \in S} \delta(x - x_i) * G_\sigma, \tag{6}$$

where D denotes the crowd density map and S is the set of all annotated points. Given a point at pixel x_i, it can be represented with a delta function $\delta(x - x_i)$. The density map can be obtained by convolving the $\delta(x - x_i)$ with Gaussian kernel with parameter G_σ. We fix Gaussian kernel size to be 15×15. The density map is 1/2 size of the original image.

Evaluation Metrics: The count error is commonly measured by two metrics, i.e., Mean Absolute Error (MAE) and Mean Squared Error (MSE),

$$\text{MAE} = \frac{1}{M} \sum_{i=1}^{M} |y_i - y_i'|, \tag{7}$$

$$\text{MSE} = \sqrt{\frac{1}{M} \sum\nolimits_{i=1}^{M} |y_i - y_i'|^2}, \tag{8}$$

and additionally GMAE,

$$\text{GMAE(L)} = \frac{1}{M} \sum_{i=1}^{M} \left(\sum_{l=1}^{4^l} |y_i^l - (y_i')^l| \right), \tag{9}$$

where M is the number of test samples, y_i is the ground truth count, and y_i' is the estimated count corresponding to the i^{th} sample. MAE indicates the accuracy of the predicted result and MSE measures the robustness. l means the region, $GMAE(L)$ divides image into a grid of 4 L non-overlapping subregions, and the error is calculated as the sum of MAE in each sub-region. Note that GMAE(0) is equivalent to MAE.

B. Datasets Details

Here we introduce four datasets, one car dataset [17], and three crowd datasets [21], [68]. The details are in Fig. 3. It can be seen that most crowd counting datasets can be regarded as complex domains due to the variable backgrounds, scale variations, and multi-level densities. And the car counting dataset is different from most crowd datasets in objects category and scenarios.

ShanghaiTech [68] is a still image dataset, with arbitrary camera perspectives and crowd density. It contains 1,198 annotated images including both internet and street view images. Part A is has 482 images, which are randomly crawled from the Internet, most of them have a large number of people. There are tremendous occlusions for most people in each image, and the scale of people is variable. As shown in Fig. 3, Part A contains moderate level density crowds.

ShanghaiTech Part B [68] is taken from the busy streets of metropolitan areas in Shanghai. ShanghaiTech Part B has 716 images, also has tremendous occlusions conditions and scale diversity in scenes, as shown in Fig. 3.

UCF-QNRF [21] is a large-scale dataset that contains a wide variety of observation viewpoints, densities, and lighting variations. The images are present in realistic scenarios captured in the wild so that they contain buildings, roads, and variable scenes. Compared with other datasets, UCF-QNRF dataset both contains high-level and low-level density, has $1,535$ images with $1, 251, 642$ annotations. We follow the setting as in [21], and split the training and test set with 1201 and 334 images, respectively. The dataset has a high average resolution of 2013×2902. Fig. 3 shows the high-density crowd.

Trancos [17] contains 1244 images of different traffic scenes collected from real video surveillance cameras. It annotates 46796 vehicles totally and provides a region of interest for each image to specify the scope of the evaluation. Some samples are shown in Fig. 3.

We do adaptations in crowd-to-car counting, which is the ShanghaiTech Part A to Trancos in Section IV-C1. In Section IV-C2, we do four types of adaptations. Here in crowd scenes, dense-to-sparse is from ShanghaiTech Part A to ShanghaiTech Part B, and sparse-to-dense is from ShanghaiTech Part B to UCF-QNRF, ShanghaiTech Part A to UCF-QNRF and ShanghaiTech Part B to ShanghaiTech Part A.

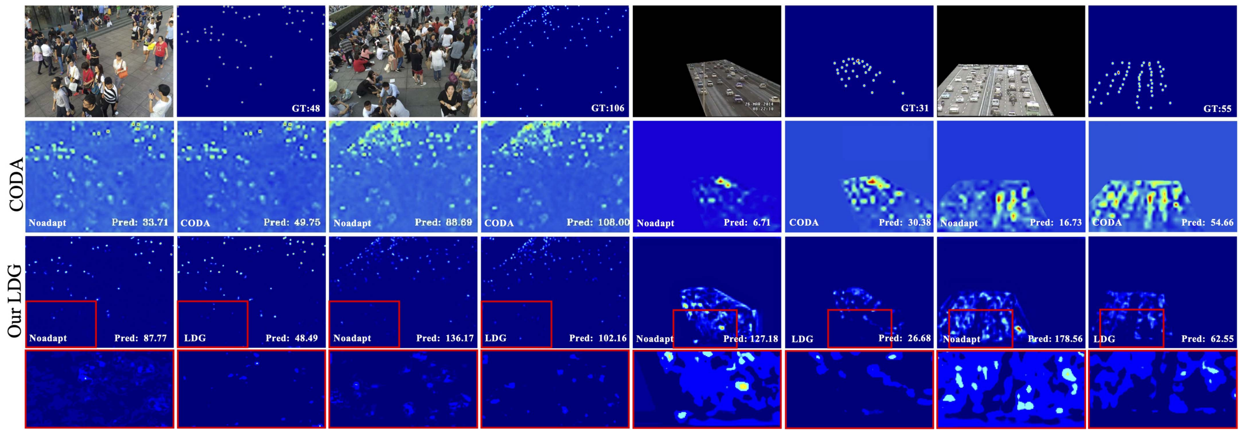

Fig. 4. The left two columns and the rigte two columns are two samples in CODA and ours, respectively. In each sample, the first row are images and ground truth; the second row is CODA noadapt and with adaptation; the third row is our LDG with noadapt and adaptation. The last row are the enlarged image with increased contrast to 90%.

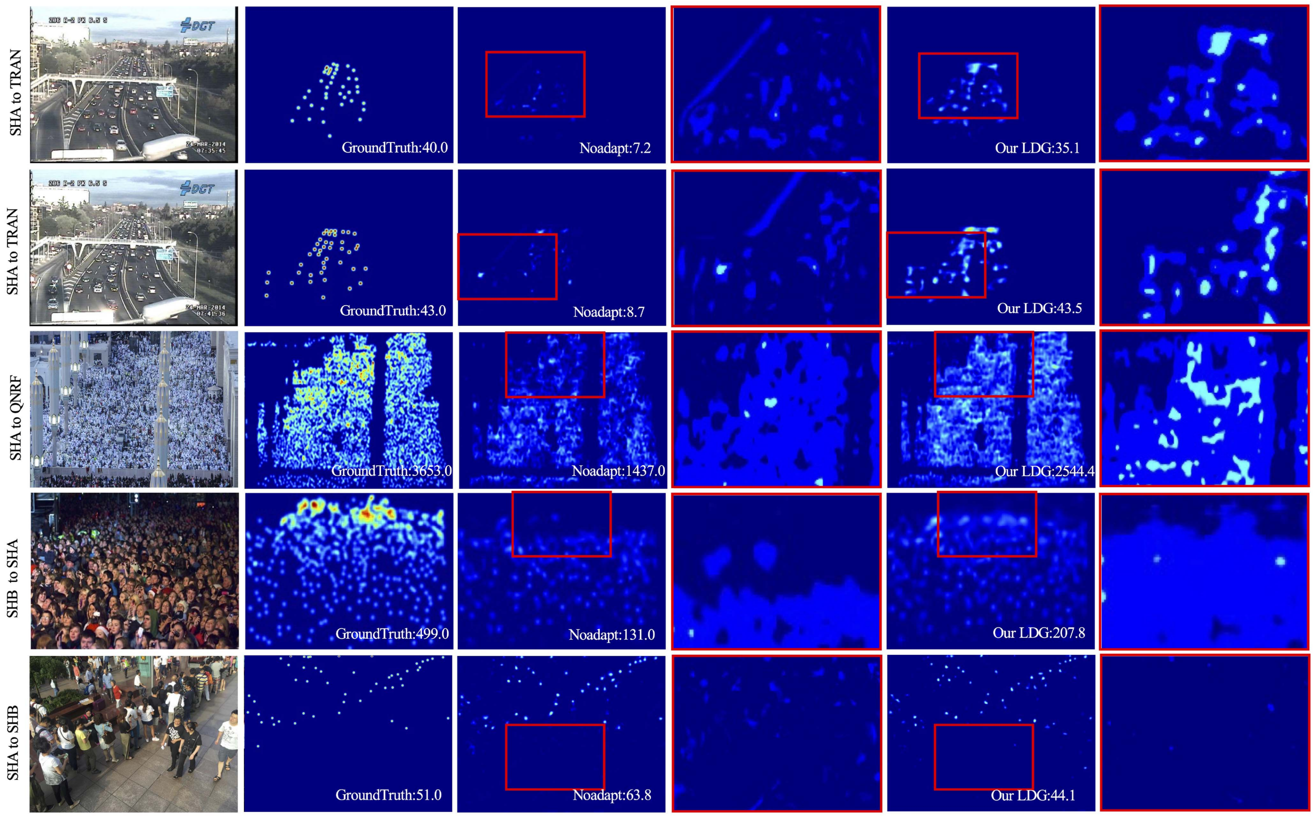

Fig. 5. Unsupervised cross-domain object counting with our Latent Domain Generation (LDG). From left to right: (1) images; (2) ground truth; (3) Without adaptation; (4) Without adaptation (contrast = 90%); (5) With our LDG; (6) With our LDG (contrast = 90%).

C. Performance and Comparison

We do experiments for unsupervised cross-domain crowd counting with our Latent Domain Generation (LDG). There are two parts in our experiments: crowd-to-cars objecting counting is in Section IV-C1 and real-to-real crowd counting in Section IV-C2.

We show the samples in Fig. 5. We can see that the quality of the unsupervised target dataset density map is significantly improved by using our LDG method.

1) Crowd-to-Car Object Counting Results: In this section, we present the adaption results across different categories, from the crowd to the cars. Trancos dataset [17] contains different traffic scenes collected from real video surveillance cameras.

Table I presents the results compared with the recent supervised state-of-the-art approaches: Lempitsky *et al.* [28], Hydra-CNN [42] and CSRNet [30]. We can see that in unsupervised domain adaptation object counting, LDG verified its effectiveness. The samples in Fig. 5 show the adaptation performance in the first two rows. After our density adaption process, our LDG

TABLE I
GMAE Comparison Results Under ShanghaiTech Part A to Trancos Setting. Here the Color blue and green Is the Best and Second Result in Recent Methods. (Note: All the Results of These Methods Are Public From These Papers)

Datasets/Metric	Supervised	Adapt	ShanghaiTech Part A [69] to Trancos [17]			
			GMAE0 ↓	GMAE1↓	GMAE2 ↓	GMAE3 ↓
Lempitsky *et al.* (NIPS2010) [28]	✓	✗	13.76	16.72	20.72	24.36
Hydra-CNN (ECCV2016) [42]	✓	✗	10.99	13.75	16.69	19.32
CSRNet (CVPR2018) [30]	✓	✗	3.56	5.49	8.57	15.04
Noadapt [29]	✗	✗	13.71	13.81	14.52	15.75
CODA [29]	✗	✓	4.91	9.89	14.88	17.55
Ours with **LDG**	✗	✓	9.00	13.09	15.78	13.09

TABLE II
Performance Comparison on Our DCNet for Real-to-Real Datasets, From the Source Domain (ShanghaiTech Part A [68]) to the Target Domains (ShanghaiTech Part B [68] and UCF-QNRF [21]). And From the Source Domain (ShanghaiTech Part B [68]) to the Target Domains (ShanghaiTech Part A [68] and UCF-QNRF [21]). Here "-" Represents No Public Results. Here the Color blue and green Is the Best and Second Result in Recent Methods. (Note: All the Results of These Methods are Public From These Papers.)

Datasets/Metric	Supervised	Adapt	A to B		A to Q		B to A		B to Q	
			MAE ↓	MSE↓	MAE↓	MSE ↓	MAE ↓	MSE↓	MAE↓	MSE ↓
MCNN (CVPR2016) [68]	✓	✗	26.4	41.3	277.0	426.0	110.2	173.2	-	-
CSRNet (CVPR2018) [30]	✓	✗	10.6	16.0	-	-	68.2	115.0	-	-
Wan *et al.* (CVPR2021) [23]	✓	✗	7.3	11.7	61.3	95.4	84.3	147.5	-	-
MCNN (CVPR2016) [68]	✗	✗	85.2	142.3	-	-	221.4	357.8	-	-
D-ConvNet-v1 (CVPR2018) [67]	✗	✗	49.1	99.2	-	-	140.4	226.1	-	-
SPN+L2SM (ICCV2019) [7]	✗	✗	21.2	38.7	227.2	405.2	126.8	203.9	-	-
RegNet (ICCV2019) [33]	✗	✗	21.6	37.5	198.7	329.4	148.9	273.8	267.2	477.6
DetNet (CVPR2019) [34]	✗	✗	55.4	90.0	411.7	731.3	242.7	400.8	411.7	731.3
CODA (ICME2019) [29]	✗	✓	15.9	26.9	-	-	-	-	-	-
Liu *et al.* (ACMMM2020) [66]	✗	✓	13.3	29.2	175.0	294.7	112.2	218.1	211.3	381.9
Ours with **LDG**	✗	✓	14.2	25.2	179.9	331.3	118.5	190.1	261.1	496.0

enhances performance in all the four GAMEs compared with Baseline. And we enhance performance up to 13.09 in GMAE3 which achieves the best result and outperforms the supervised methods [28], [30], [42]. We also get competitive results in the other three GAME metrics. All the results show that our LDG presents great effectiveness on adaption across categories.

2) Real-to-Real Crowd Counting Results: In real-to-real adaptation, we do four types of adaptations. Here in crowd scenes, dense-to-sparse is from ShanghaiTech A to ShanghaiTech B (A to B), and sparse-to-dense is from ShanghaiTech B to UCF-QNRF (B to Q), ShanghaiTech A to UCF-QNRF (A to Q), and ShanghaiTech B to ShanghaiTech A (B to A). These results are shown in Table II.

Dense-to-sparse Crowd Counting: From ShanghaiTech A to ShanghaiTech B (A to B) adaptation, Table II presents the results in both supervised methods and unsupervised methods. We report performances of the state-of-the-art full-supervised approaches, including MCNN [68], Wan *et al.* [23] and CSRNet [30]. We also show the performance of some works with no adaptation [7], [34], [67], [68] in cross-dataset. Compared with other unsupervised adaptation methods without any annotations on the target dataset, our LDG achieves competitive MAE and MSE results in [7], [29], [34], [66]–[68] in all adaptation settings. For the dense-to-sparse (A to B) adaptation, our LDG gets the best MSE and the second MAE in all methods. In addition, the quality of our density map is very closer to the ground truth compared to CODA [29] as shown in Fig. 4.

Sparse-to-dense Crowd Counting: For the sparse-to-dense adaptation, the performance in the adaptations of ShanghaiTech B to UCF-QNRF (B to Q), ShanghaiTech A to UCF-QNRF (A to Q), and ShanghaiTech B to ShanghaiTech A (B to A) is competitive. Most of our adaptation results get the best and second performance in terms of MAE and MSE compared with other methods [7], [29], [34], [66]–[68]. Our LDG gets the best performance of MSE in A to B and B to A, and gets the second performance of MAE in all results. Fig. 5 presents some example results for adapted density maps. We can see that our LDG improves the quality of density maps compared with the result without adaptation.

D. Ablation Study

We first briefly show the benefit of applying the adaptation module and consistency module in Section IV-D1. Then we analyze the effectiveness of our latent domain generator in Section IV-D2, comparing it with noise embedding and vanilla auto-encoder. We show the analysis for architectures and hyperparameters in Section IV-D3 and Section IV-D4.

1) Effectiveness of Different Modules: Here we prove the benefit of each component of LDG, including the adaptation module and consistency module. Results are shown in Table IV. We can see that our LDG gets the best performance in all adaptation settings, which demonstrates that the proposed adversarial and consistency loss with our domain generator are effective. With our *Domain Generator* (DG), the adaptation with only adversarial loss performs worse (from 118.5 to 136.6 in terms of MAE) than the proposed LDG due to that

TABLE III
COMPARISON ON DIFFERENT ARCHITECTURES OVER HOURGLASS [41] AND VGG-16 [49]. THERE ARE TWO TASKS SETTINGS, SUPERVISED LEARNING ON SHANGHAITECH PART A [68], AND UNSUPERVISED DOMAIN ADAPTATION FROM THE SOURCE DOMAIN (SHANGHAITECH PART A [68]) TO THE TARGET DOMAINS (SHANGHAITECH PART B [68] AND UCF-QNRF [21])

Methods/Metric	Parameters	ShanghaiTech Part A [69]		A to B		A to Q	
		MAE ↓	MSE ↓	MAE ↓	MSE ↓	MAE ↓	MSE ↓
VGG-16 [49]	13.8M	71.4	125.4	14.2	25.2	179.9	331.3
Stacked Hourglass [41]	13.3M	66.7	106.9	18.5	28.7	171.0	344.9

TABLE IV
ABLATION STUDY ON DIFFERENT MODULES. ARROWS IN ALL TABLES INDICATE THE FAVORABLE DIRECTIONS OF THE METRIC VALUES. THIS IS IN THE ADAPTATION FROM THE SOURCE DOMAIN (SHANGHAITECH PART B [68]) TO THE TARGET DOMAIN (SHANGHAITECH PART A [68]). BEST PERFORMANCE IN EACH GROUP IS **BOLD**

Configurations	Settings	Metrics	
		MAE↓	MSE↓
w/o DG	NoAdapt	140.0	202.3
	Vanilla Adapt	126.4	205.3
w/ DG	Adapt with latent domain by $\mathcal{L}_{adv}$	136.6	217.8
	Adapt with latent domain by $\mathcal{L}_{con}$	122.7	192.8
	Ours with **LDG**	**118.5**	**190.1**

TABLE V
COMPARISON OF DIFFERENT COMPONENTS IN THE ADAPTATION MODULE. THIS IS IN THE ADAPTATION B TO A, FROM THE SOURCE DOMAIN (SHANGHAITECH PART B [68]) TO THE TARGET DOMAIN (SHANGHAITECH PART A [68]). RANDOM NOISE IS FROM A STANDORD DISTRIBUTION $n \sim N(0,1)$. VANILLA AUTO-ENCODER IS AN ENCODER-DECODER WITHOUT NOISE

Datasets	B to A	
Metric	MAE↓	MSE↓
NoAdapt	140.0	202.3
Ours with **LDG**	**118.5**	**190.1**
Our Adapt *w/* DG (Random Noise)	122.7	198.2
Our Adapt *w/* DG (Vanilla Auto-encoder)	135.4	212.3

TABLE VI
DETAILED ARCHITECTURE OF LDG. WHERE CONV. (A, B, C) REPRESENTS THE CONVOLUTION OPERATION OF KERNEL SIZE = A*B AND OUPUT CHANNEL = C. BN REPRESENTS THE BATCH NORM OPERATION. RELU, LEAKY-RELU AND SIGMOD REPRESENT THE ACTIVATION FUNCTION

Feature Generator	
[Layer 1]	Conv. (3, 3, 64); ReLU; Conv. (3, 3, 64); ReLU; MaxPool2d (2, 2), stride=2;
[Layer 2]	Conv. (3, 3, 128); ReLU; Conv. (3, 3, 128); ReLU; MaxPool2d (2, 2), stride=2;
[Layer 3]	Conv. (3, 3, 256); ReLU; Conv. (3, 3, 256); ReLU; Conv. (3, 3, 256); ReLU; MaxPool2d (2, 2), stride=2;
[Layer 4]	Conv. (3, 3, 512); ReLU; Conv. (3, 3, 512); ReLU; Conv. (3, 3, 512); ReLU;
Domain Generator	
[Layer 1]	Conv. (3, 3, 64); BN; ReLU; MaxPool2d (2, 2), stride=2;
[Layer 2]	Conv. (3, 3, 256); BN; ReLU; MaxPool2d (2, 2), stride=2;
[Layer 3]	BN; ReLU; Conv. (1, 1, 128); BN; ReLU; Conv. (3, 3, 128); BN; ReLU; Conv. (1, 1, 256); Residual connection;
[Layer 4]	BN; ReLU; Conv. (1, 1, 128); BN; ReLU; Conv. (3, 3, 128); BN; ReLU; Conv. (1, 1, 256); Residual connection;
[Layer 5]	BN; ReLU; Conv. (1, 1, 128); BN; ReLU; Conv. (3, 3, 128); BN; ReLU; Conv. (1, 1, 256); Residual connection;
[Layer 6]	Conv. (3, 3, 64); BN; ReLU; Upsample (scale_factor=2);
[Layer 7]	Conv. (3, 3, 3); BN; ReLU; Upsample (scale_factor=2);
Adversarial Discriminator	
[Layer 1]	Conv. (4, 4, 64), stride=2; Conv. (4, 4, 128), stride=2; Conv. (4, 4, 256), stride=2;Conv. (4, 4, 512), stride=2;
[Layer 2]	Conv. (4, 4, 1), stride=2; Leaky-ReLU;
Density Decoder	
[Layer 1]	Upsample (scale_factor=4); Conv. (1, 1, 128); ReLU; Conv. (1, 1, 1); ReLU;

there is no constraint for DG from the source domain. While the adaptation only with consistency loss also performs worse than LDG (MAE/MSE: 118.5/190.1 v.s. 122.7/192.8) because of unaligned distributions. Furthermore, if we finally drop out the domain generator, our LDG degenerates into the *vanilla adapt*, a universal adversarial training method in unsupervised domain adaptation, like CODA [29] in crowd-to-car counting. Our LDG surpasses the *vanilla adapt* results, and achieves the best performance. The experiments demonstrate the effectiveness of our LDG with the proposed adaptation and consistency module.

2) Effectiveness of Latent Domain Generator: Our adaptation module consists of a domain generator, where the domain generator is an auto-encoder with random perturbations. Here we show the effectiveness of the designed domain generator in this part. The simplest choice is adding the random noise perturbation to the source domain as shown in Table V. We also show the results of taking the vanilla auto-encoder as the domain generator, which is a simple encoder-decoder without noise, as shown in the last row in Table V. The results of *random noise* and *vanilla auto-encoder* show some improvement compared to the baseline (NoAdapt). While our LDG is the way to combine the random characteristics with the target information in an encoder-decoder network. Thus, the improvement of our domain generator is reasonable due to the random characteristics bringing perturbations to the model, while the auto-encoder structure further extracts the target information with perturbations into a latent distribution in a generated domain space.

3) Comparison on Different Architectures: In this part, we re-examine the choice of different architectures, and conduct experiments over VGG-16 [49] and Hourglass [41], where VGG-16 [49] is commonly used in recent state-of-the-art supervised/unsupervised counting benchmarks (*e.g.*, CSRNet [30], CAN [36], CODA [29], SE-CycleGAN [59]) and Hourglass [41] is a high-resolution-friendly architecture with the details for images. We employ the Hourglass architecture in [41] by stacking two hourglasses, which is with comparable parameters (13.3 M) with VGG-16 (13.8 M). The feature encoder and density decoder are from the front-backbone after all the skipping connections and the last convolutional layer of stacked hourglasses, respectively. The stacked Hourglass is trained under the same settings of training LDG for fair comparison, as shown in Table III. Firstly, in supervised learning setting, Hourglass surpasses the VGG-16 in SHA dataset in terms of MAE and MSE (66.7/106.9 v.s. 71.4/125.4). In the unsupervised domain adaptation experiments, the Hourglass can hardly surpass the backbone

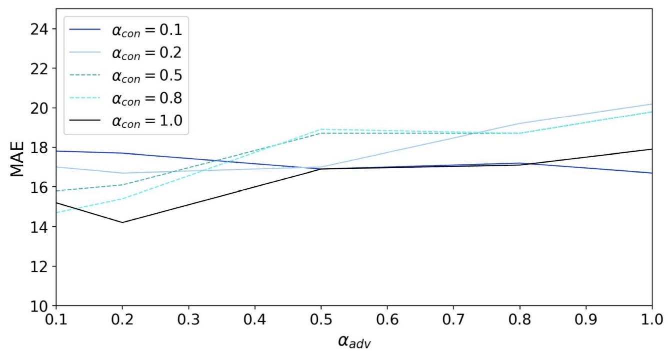

Fig. 6. Illustration of performance comparison when we set different values α_{con} and α_{adv} for $\mathcal{L}_{con}$ and $\mathcal{L}_{adv}$, in the adaptation from the source domain (ShanghaiTech Part A [68]) to the target domain (ShanghaiTech Part B [68]).

VGG-16 in terms of all metrics, but could also get competitive results compared with recent state-of-the-art methods in Table II. Meanwhile, the performance of Hourglass architecture does not drop significantly compared with VGG-16 in our LDG, which demonstrates the robustness of the LDG and reflects that our method is not very sensitive to the base architectures.

4) Analysis of the Hyper-Parameters: To verify the choice of hyper-parameters for the proposed losses, we conduct experiments to evaluate their performance under different values of them, in the adaptation from ShanghaiTech Part A [68] to Part B [68]. The consistency loss is used to regularize the source domain and latent domain to avoid uncontrollable noise domain far away from the original source domain, and the adversarial loss can narrow the discrepancy between the latent domain and target domain.As shown in Fig. 6, LDG gets the best results when the weight of the consistency and adversarial loss as 1.0 and 0.2, respectively. Meanwhile, the counting accuracy does not drop significantly in some conditions with different weights of $\mathcal{L}_{adv}$ and $\mathcal{L}_{con}$, which demonstrates the robustness of our LDG.

V. Conclusion

In this paper, we propose a latent domain generation method for unsupervised object counting, which tackles the problems of complex domain shifts across domains. The domain generator can generate latent domain distribution with random perturbations from the target domain. At the same time, under the constraint of consistency loss, the robustness of the model to different distributions can be guaranteed. Experiments have proved that compared with the previous methods, aligning the distribution of the generated domain and target domain with the adversarial learning improves the performance of unsupervised domain adaptation objects counting.

Appendix

In this Appendix, we provide the Detailed Architecture of LDG. It introduces the detailed architecture of four blocks in our Latent Domain Generation (LDG) network: Feature Generator, Domain Generator, Adversarial Discriminator, and Density Decoder.

Feature Generator is constructed with the first 13 convolution layers from VGG-16 [49] in realistic-to-realistic. The output size of VGG-16 Feature Generator is $[Batch, 512, H/8, W/8]$

The Domain Generator contains some convolution layers, which is an auto-encoder architecture. The encoder of Domain Generator contains two convolutional layers kernel size 3×3, the channels of each layer are $\{64, 256\}$ respectively. Each convolutional layer adds the maxpooling operation. Then we apply some residual convolutional layers in channel 256 and kernel size 3. And the decoder does the operation to restore the image size corresponding to the input.

The Adversarial Discriminator contains five convolutional layers with stride of 2 and kernel size 4×4, the channels of each layer are $\{64, 128, 256, 512, 1\}$ respectively.

The Density Decoder increases the density size comfortably to the ground truth.

References

[1] S. Bai, Z. He, Y. Qiao, H. Hu, and J. Yan, “Adaptive dilated network with self-correction supervision for counting,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4593–4602.
[2] L. Boominathan, S. S. Kruthiventi, and R. V. Babu, “CrowdNet: A deep convolutional network for dense crowd counting,” in *Proc. ACM Int. Conf. Multimedia*, 2016, pp. 640–644.
[3] X. Cao, Z. Wang, Y. Zhao, and F. Su, “Scale aggregation network for accurate and efficient crowd counting,” in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 734–750.
[4] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos, “Privacy preserving crowd monitoring: Counting people without people models or tracking,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–7.
[5] C. Change Loy, S. Gong, and T. Xiang, “From semi-supervised to transfer counting of crowds,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2256–2263.
[6] K. Chen, C. C. Loy, S. Gong, and T. Xiang, “Feature mining for localised crowd counting,” in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–11.
[7] X. Chenfeng *et al.*, “Learn to scale: Generating multipolar normalized density maps for crowd counting,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 8382–8390.
[8] H. Cholakkal, G. Sun, F. S. Khan, and L. Shao, “Object counting and instance segmentation with image-level supervision,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12389–12397.
[9] A. Coates, A. Ng, and H. Lee, “An analysis of single-layer networks in unsupervised feature learning,” in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 215–223.
[10] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 886–893.
[11] W. Deng *et al.*, “Informative feature disentanglement for unsupervised domain adaptation,” *IEEE Trans. Multimedia*, early access, May 21, 2021, doi: 10.1109/TMM.2021.3080516.
[12] Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by backpropagation,” in *Proc. Int. conf. mach. learn.*, 2015, pp. 1180–1189.
[13] J. Gao, Y. Yuan, and Q. Wang, “Feature-aware adaptation and density alignment for crowd counting in video surveillance,” *IEEE Trans. Cybern.*, vol. 51, no. 10, pp. 4822–4833, Oct. 2021.
[14] B. Gong, Y. Shi, F. Sha, and K. Grauman, “Geodesic flow kernel for unsupervised domain adaptation,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2066–2073.
[15] R. Gong, W. Li, Y. Chen, and L. V. Gool, “Dlow: Domain flow for adaptation and generalization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2477–2486.
[16] R. Gopalan, R. Li, and R. Chellappa, “Domain adaptation for object recognition: An unsupervised approach,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 999–1006.
[17] R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, and R. López-Sastre, “Extremely overlapping vehicle counting,” in *Proc. Iberian Conf. Pattern Recognit. Image Anal.*, 2015, pp. 423–431.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.
[19] J. Hoffman *et al.*, "Cycada: Cycle-consistent adversarial domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1989–1998.
[20] H. Idrees, I. Saleemi, C. Seibert, and M. Shah, "Multi-source multi-scale counting in extremely dense crowd images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 2547–2554.
[21] H. Idrees *et al.*, "Composition loss for counting, density map estimation and localization in dense crowds," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 532–546.
[22] S. Jegou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1175–1183.
[23] W. Jia, L. Ziquan, and B. C. Antoni, "A generalized loss function for crowd counting and localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1974–1983.
[24] X. Jiang *et al.*, "Density-aware multi-task learning for crowd counting," *IEEE Trans. Multimedia*, vol. 23, pp. 443–453, 2021.
[25] M. Kim and H. Byun, "Learning texture invariant representation for domain adaptation of semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12975–12984.
[26] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013, *arXiv:1312.6114*.
[27] D. Kong, D. Gray, and H. Tao, "A viewpoint invariant approach for crowd counting," in *Proc. Int. Conf. Pattern Recognit.*, 2006, pp. 1187–1190.
[28] V. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1324–1332.
[29] W. Li, L. Yongbo, and X. Xiangyang, "Coda: Counting objects via scale-aware adversarial density adaption," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2019, pp. 193–198.
[30] Y. Li, X. Zhang, and D. Chen, "CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1091–1100.
[31] C. Liu, X. Weng, and Y. Mu, "Recurrent attentive zooming for joint crowd counting and precise localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1217–1226.
[32] L. Liu *et al.*, "DENet: A universal network for counting crowd with varying densities and scales," *IEEE Trans. Multimedia*, vol. 23, pp. 1060–1068, 2021.
[33] L. Liu *et al.*, "Crowd counting with deep structured scale integration network," in *Proc. IEEE Conf. Comput. Vis.*, 2019, pp. 1774–1783.
[34] W. Liu, S. Liao, W. Ren, W. Hu, and Y. Yu, "High-level semantic feature detection: A new perspective for pedestrian detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5187–5196.
[35] W. Liu, M. Salzmann, and P. Fua, "Context-aware crowd counting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5094–5103.
[36] W. Liu, M. Salzmann, and P. Fua, "Context-aware crowd counting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5099–5108.
[37] Y. Liu, M. Shi, Q. Zhao, and X. Wang, "Point in, box out: Beyond counting persons in crowds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6469–6478.
[38] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.
[39] X. Ma, T. Zhang, and C. Xu, "Deep multi-modality adversarial networks for unsupervised domain adaptation," *IEEE Trans. Multimedia*, vol. 21, no. 9, pp. 2419–2431, Sep. 2019.
[40] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Proc. Int. Conf. Artif. Neural Netw.*, 2011, pp. 52–59.
[41] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 483–499.
[42] D. Onoro-Rubio and R. J. López-Sastre, "Towards perspective-free object counting with deep learning," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 615–629.
[43] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 53–69, May 2015.
[44] M. K. K. Reddy, M. A. Hossain, M. Rochan, and Y. Wang, "Few-shot scene adaptive crowd counting using meta-learning," in *Proc. IEEE Int. Conf. Appl. Comput. Vis.*, 2020, pp. 2803–2812.
[45] D. Ryan, S. Denman, S. Sridharan, and C. Fookes, "An evaluation of crowd counting methods, features and regression models," *Comput. Vis. Image Understanding*, vol. 130, pp. 1–17, 2015.
[46] S. A. M. Saleh, S. A. Suandi, and H. Ibrahim, "Recent survey on crowd density estimation and counting for visual surveillance," *Eng. Appl. Artif. Intell.*, vol. 41, pp. 103–114, 2015.
[47] D. B. Sam and R. V. Babu, "Top-down feedback for crowd counting convolutional neural network," in *Proc. Assoc. Advance. Artif. Intell.*, 2018, pp. 7323–7330.
[48] L. Samuli and A. Timo, "Temporal ensembling for semi-supervised learning," 2016, *arXiv:1610.02242*.
[49] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[50] V. A. Sindagi and V. M. Patel, "Generating high-quality crowd density maps using contextual pyramid CNNs," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1879–1888.
[51] V. A. Sindagi, R. Yasarla, D. S. Babu, R. V. Babu, and V. M. Patel, "Learning to count in the crowd from limited labeled data," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 212–229.
[52] B. Tan, J. Zhang, and L. Wang, "Semi-supervised elastic net for pedestrian counting," *Pattern Recognit.*, vol. 44, no. 10–11, pp. 2297–2304, 2011.
[53] Y.-H. Tsai *et al.*, "Learning to adapt structured output space for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7472–7481.
[54] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.
[55] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp 137–154, 2004.
[56] T.-H. Vu, H. Jain, M. Bucher, M. Cord, and P. Pérez, "Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2517–2526.
[57] J. Wan, W. Luo, B. Wu, A. B. Chan, and W. Liu, "Residual regression with semantic prior for crowd counting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4031–4040.
[58] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 2018.
[59] Q. Wang, J. Gao, W. Lin, and Y. Yuan, "Learning from synthetic data for crowd counting in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8198–8207.
[60] Q. Wang, J. Gao, W. Lin, and Y. Yuan, "Pixel-wise crowd understanding via synthetic data," *Int. J. Comput. Vis.*, vol. 129, pp. 1–21, 2020.
[61] Q. Wang, T. Han, J. Gao, and Y. Yuan, "Neuron linear transformation: Modeling the domain shift for crowd counting," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 27, 2021, doi: 10.1109/TNNLS.2021.3051371.
[62] R. Wang, "Cross-domain contrastive learning for unsupervised domain adaptation," *IEEE Trans. Multimedia*, early access, Jan. 27, 2022, doi: 10.1109/TMM.2022.3146744.
[63] Z. Yan *et al.*, "Perspective-guided convolution networks for crowd counting," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 952–961.
[64] F. Yang *et al.*, "Part-aware progressive unsupervised domain adaptation for person re-identification," *IEEE Trans. Multimedia*, vol. 23, pp. 1681–1695, 2021.
[65] G. Yang, H. Xia, M. Ding, and Z. Ding, "Bi-directional generation for unsupervised domain adaptation," *Assoc. Advance. Artif. Intell.*, vol. 34, pp. 6615–6622, 2020.
[66] L. Yuting *et al.*, "Towards unsupervised crowd counting via regression-detection bi-knowledge transfer," in *Proc. 28th ACM Int. Conf. Multimedia*, 2020, pp. 129–137.
[67] S. Zan, X. Yi, B. Ni, M. Wang, and X. Yang, "Crowd counting via adversarial cross-scale consistency pursuit," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5245–5254.
[68] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 589–597.
[69] Y. Zou, Z. Yu, B. V. Kumar, and J. Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 289–305.