Neural Networks 184 (2025) 107031

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Full Length Article

Domain-guided conditional diffusion model for unsupervised domain adaptation

Yulong Zhang [a,1,2], Shuhao Chen [b,1], Weisen Jiang [b,c], Yu Zhang [b,*], Jiangang Lu [a], James T. Kwok [c]

[a] State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou, 310027, Zhejiang, China
[b] Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, 518055, Guangdong, China
[c] Department of Computer Science and Engineering, Hong Kong University of Science and Technology, 999077, Hong Kong, China

ARTICLE INFO

Keywords:
Diffusion models
Transfer learning
Unsupervised domain adaptation

ABSTRACT

Limited transferability hinders the performance of a well-trained deep learning model when applied to new application scenarios. Recently, Unsupervised Domain Adaptation (UDA) has achieved significant progress in addressing this issue via learning domain-invariant features. However, the performance of existing UDA methods is constrained by the possibly large domain shift and limited target domain data. To alleviate these issues, we propose a Domain-guided Conditional Diffusion Model (DCDM), which generates high-fidelity target domain samples, making the transfer from source domain to target domain easier. DCDM introduces class information to control labels of the generated samples, and a domain classifier to guide the generated samples towards the target domain. Extensive experiments on various benchmarks demonstrate that DCDM brings a large performance improvement to UDA.

1. Introduction

Deep neural networks have achieved remarkable advances in a variety of applications due to their powerful representation learning capabilities (Li et al., 2024; Villaizán-Vallelado, Salvatori, Carro, & Sanchez-Esguevillas, 2024; Wang, 2024; Zhang et al., 2022; Zhang & Yang, 2022). However, when domain shift occurs, a model trained from a source domain may suffer noticeable performance degradation in the target domain Oza, Sindagi, Sharmini, and Patel (2023), Pan and Yang (2009). To handle this issue, Unsupervised Domain Adaptation (UDA) (Fang, Yap, Lin, Zhu, & Liu, 2024; Gu, Sun, & Xu, 2022; Yang, Zhang, Dai, & Pan, 2020; Zhang, Yao, Chen, Jin, Jin, & Jiangang, 2024; Zhao et al., 2022) is proposed to learn knowledge from the source domain as well as unlabeled target domain, and then transfer the knowledge to help learn from the target domain.

To alleviate domain shift, many UDA methods (Dhaini, Berar, Honeine, & Van Exem, 2023; Li, Lu, Zuo, & Zhang, 2022; Long, Cao, Wang, & Jordan, 2015; Rangwani, Aithal, Mishra, Jain, & Radhakrishnan, 2022; Wang et al., 2023) are proposed to learn domain-invariant features explicitly or implicitly. Specifically, Long et al. (2015), Zhang, Liu, Long, and Jordan (2019), and Zhu et al. (2021) explicitly minimize the distribution discrepancy between the source and target domains. On the other hand, adversarial-based UDA methods enhance transfer learning capabilities by confusing the source and target domain features (Chen, Wu, Duan, & Gao, 2022; Rangwani et al., 2022; Zhang et al., 2023) or generating auxiliary samples to bridge the source and target domains (Hoffman et al., 2018; Liu et al., 2024; Yang, Xia, Ding & Ding, 2020). However, the performance of existing UDA methods is constrained by the unlabeled target domain. When there is a large domain shift across domains, it further poses challenges to the UDA methods. Moreover, when the target domain samples are limited,[3] it is difficult for UDA methods to accurately model the target domain's data distribution. As a result, direct domain alignment between the source samples and limited target domain samples can cause sub-optimal transfer effects. Different from existing mainstream UDA methods (including discrepancy-based and adversarial-based methods) that focus on aligning the distributions between the source and target

* Corresponding author.
E-mail addresses: zhangylcse@zju.edu.cn (Y. Zhang), 12232388@mail.sustech.edu.cn (S. Chen), wjiangar@cse.ust.hk (W. Jiang), yu.zhang.ust@gmail.com (Y. Zhang), lujg@zju.edu.cn (J. Lu), jamesk@cse.ust.hk (J.T. Kwok).

[1] Equal contribution.

[2] This work was done when the first author is a visiting student at Southern University of Science and Technology.

[3] For example, in the experiments, the Dslr and Webcam domains in the *Office31* dataset have an average of only 16 and 26 samples per class, respectively. The Art domain in the *Office–Home* dataset has an average of 37 samples per class.

https://doi.org/10.1016/j.neunet.2024.107031
Received 29 July 2024; Received in revised form 17 November 2024; Accepted 6 December 2024
Available online 26 December 2024

domains, augmentation-based UDA methods generate more data for training, which is beneficial to UDA methods (Na, Jung, Chang, & Hwang, 2021; Pan & Yang, 2009; Shorten & Khoshgoftaar, 2019). For example, Chen et al. (2023) and Gao, Chu, Wang, and Stankovic (2022) generate source/intermediate domain images, while Hoffman et al. (2018) and Yang, Xia, et al. (2020) employ image-to-image translation to bridge the source and target domains. However, the above methods cannot generate samples conditioned on labels, making it difficult to model the target domain distribution. Moreover, as demonstrated in Appendix, we find that incorporating labeled target samples into the source domain can reduce the generalization error bound on the target domain.

To achieve that, in this paper, we propose the Domain-guided Conditional Diffusion Model (DCDM) to generate "labeled" target domain samples for UDA methods. Using the Diffusion Probabilistic Models (DPM) (Cao et al., 2024; Dhariwal & Nichol, 2021; Ho, Jain, & Abbeel, 2020) (or simply diffusion models), DCDM can control the class and domain of each generated sample. Specifically, to control the class of each generated sample, we inject label information into the adaptive group normalization (AdaGN) layers (Dhariwal & Nichol, 2021). For the source domain samples, we simply use their ground truth labels, while for the unlabeled target domain samples, we use pseudo-labels as predicted by a pretrained UDA model. To control the domain of each generated sample, we train a domain classifier to guide the diffusion sampling process towards the target domain in the generation. The generated target domain samples are then combined with the source domain samples as an augmented source domain, which is closer to the target domain than the original source domain, making it easier for UDA methods to perform transfer learning. Finally, we train a UDA model to transfer from the augmented source domain to the target domain.

The proposed DCDM is very flexible and can be used with any existing UDA method. We analyze theoretically its generalization bound. Empirically, extensive experiments on various UDA benchmarks demonstrate the superiority of DCDM.

The contributions of this paper are four-fold.

(1) We manage to control domain and class generation through domain guidance and class condition, respectively, and propose a novel DCDM framework that generates high-fidelity samples for the unlabeled target domain. To the best of our knowledge, this is the first application of DPMs for UDA.
(2) DCDM can be used with any existing UDA method. Specifically, in our experiments, we combine DCDM with MCC (Jin, Wang, Long, & Wang, 2020), ELS (Zhang et al., 2023), and SSRT (Sun, Lu, Zhang, & Ling, 2022) to achieve state-of-the-art performance on various benchmark datasets.
(3) Theoretically, we establish a generalization bound of DCDM.
(4) Empirically, we perform extensive experiments on UDA benchmark datasets to demonstrate the effectiveness of DCDM. Results show that the generated samples are similar to the target domain samples, and the augmented source domain can help reduce domain shift.

The rest of this paper is organized as follows. Section 2 reviews related studies. Section 3 details the proposed DCDM framework and establishes a generalization bound for DCDM. Section 4 shows experimental results on UDA benchmarks, and finally Section 5 concludes the paper.

2. Related work

2.1. Unsupervised Domain Adaptation

UDA methods (Ganin et al., 2016; Liu et al., 2023; Zhao et al., 2022; Zhuang, Zhang, & Wei, 2024) extract knowledge from the labeled source domain to facilitate learning in the unlabeled target domain. As the data distribution of the target domain differs from that of the source domain, various methods have been proposed to reduce domain discrepancy. They can mainly be classified into three categories: discrepancy-based methods, adversarial-based methods, and augmentation-based methods.

Discrepancy-based methods learn a feature extractor to minimize the distribution discrepancy between the source and target domains. For example, DAN (Long et al., 2015) minimizes the maximum mean discrepancy (MMD) (Gretton, Borgwardt, Rasch, Schölkopf, & Smola, 2012) between domains, while deep subdomain adaptation network (DSAN) (Zhu et al., 2021) considers the relationship between two subdomains within the same class but across different domains. Margin disparity discrepancy (MDD) (Zhang et al., 2019) performs domain alignment with generalization error analysis. Adaptive feature norm (AFN) (Xu, Li, Yang, & Lin, 2019) progressively adapts the feature norms of the source and target domains to improve transfer performance. Unlike the above methods that explicitly align domains, minimum class confusion (MCC) (Jin et al., 2020) introduces a general class confusion loss as a regularizer.

Adversarial-based methods align the source and target distributions through adversarial training (Goodfellow, Shlens, & Szegedy, 2015). For example, domain adversarial neural network (DANN) (Ganin et al., 2016) learns a domain discriminator to distinguish samples in the two domains, and uses a feature generator to confuse the domain discriminator. Conditional domain adversarial network (CDAN) (Long, Cao, Wang, & Jordan, 2018) injects class-specific information into the discriminator to facilitate alignment of the multi-modal distributions. Smooth domain adversarial training (SDAT) (Rangwani et al., 2022) employs sharpness-aware minimization (Foret, Kleiner, Mobahi, & Neyshabur, 2021) to seek a flat minimum for better generalization. Environment label smoothing (ELS) (Zhang et al., 2023) alleviates the effect of label noise by encouraging the domain discriminator to output soft domain labels. Sun et al. (2022) introduce a Vision Transformer (ViT) (Dosovitskiy, 2020) backbone and a safe self-refinement strategy to enhance model performance while mitigating the risk of collapse in UDA. Transferable vision transformer (TVT) (Yang, Liu, Xu & Huang, 2023) leverages ViT for UDA by injecting learned transferabilities into the attention blocks. Huang, Song, and Zhang (2024) propose gradient harmonization techniques to resolve optimization conflicts between domain alignment and classification by adjusting gradient angles. Litrico, Del Bue, and Morerio (2023) refine pseudo-labels through uncertainty estimation and knowledge aggregation from neighboring samples to improve performance in source-free UDA.

Augmentation-based methods bridge the source and target domains by data augmentation. For instance, cycle-consistent adversarial domain adaptation (CyCADA) (Hoffman et al., 2018) uses the generative adversarial network (GAN) for image-to-image translation via the cycle consistency loss and semantic consistency loss. Bi-directional generation (BDG) (Yang, Xia, et al., 2020) uses two cross-domain generators to synthesize data of each domain conditioned on the other, and learns two task-specific classifiers. Adapt anything (Chen et al., 2023) is a task-agnostic method that leverages the prior knowledge of Stable Diffusion (Rombach, Blattmann, Lorenz, Esser, & Ommer, 2022) to synthesize surrogate source data. While this method avoids the need to fine-tune the diffusion model, it relies heavily on the prior knowledge of the pre-trained model for specific domains and categories. Unlike existing augmentation-based methods that generate source/intermediate domain images (Chen et al., 2023; Gao et al., 2022) or employ image-to-image translation (Hoffman et al., 2018; Yang, Xia, et al., 2020), the proposed method generates "labeled" target domain samples for the unlabeled target domain, which is challenging. Those "labeled" samples can bring extra supervised signals for training the UDA model.

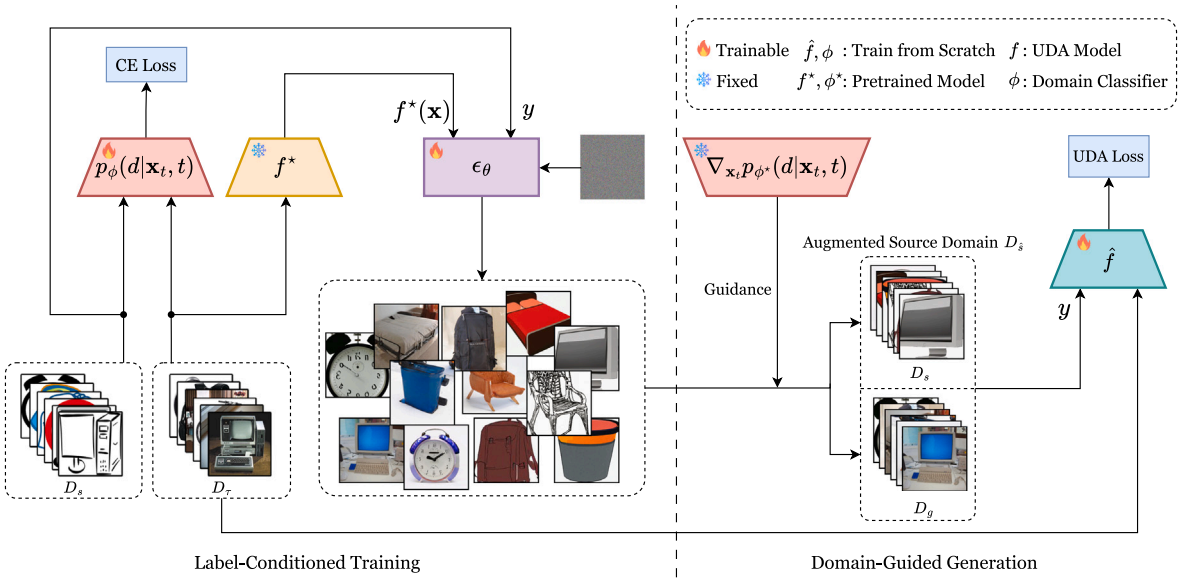


Fig. 1. An overview of the Domain-guided Conditional Diffusion Model (DCDM). In *label-conditioned training*, a conditional diffusion model ϵ_θ is trained on the source domain samples (with ground-truth labels) and target domain samples (with pseudo-labels predicted by the pretrained classifier $f^\star$). $\epsilon_\theta^\star$ denotes the optimized ϵ_θ. A domain classifier $\boldsymbol{\phi}$ is trained to discriminate samples from the source and target domains. In *domain-guided generation*, the generated target domain data D_g are sampled from the trained $\epsilon_\theta^\star$ with guidance provided by the domain classifier $\boldsymbol{\phi}^\star$. Finally, a UDA model $\hat{f}$ is trained to transfer from the augmented source domain $D_{\hat{s}} = D_s \cup D_g$ to the target domain D_τ.

2.2. Diffusion probabilistic model (DPM)

In recent years, the DPM (Sohl-Dickstein, Weiss, Maheswaranathan, & Ganguli, 2015) has achieved great success in various generative tasks (Wang, Wu, Yuan, Tong, & Xu, 2024; Ye & Liu, 2024). It includes a forward diffusion process that converts an original sample $\mathbf{x}_0$ to a Gaussian noise, and a reverse denoising process that infers the Gaussian noise back to a sample. Specifically, the forward process gradually injects noise to $\mathbf{x}_0$ until it becomes a random noise $\mathbf{x}_T$, via the sampling: $q\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right) = \mathcal{N}\left(\sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I}\right)$, where $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, and α_t's follow a decreasing schedule. On the other hand, the reverse process denoises $\mathbf{x}_T$ to $\mathbf{x}_0$ using a decoder $p_\theta\left(\mathbf{x}_{t-1}|\mathbf{x}_t\right)$. Specifically, a model $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t\right)$, parameterized by $\boldsymbol{\theta}$, is trained to predict the noise injected in the forward process by minimizing an approximate loss

$$\mathbb{E}_{t\sim\mathcal{U}(1,T),\mathbf{x}_0,\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\omega_t\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t,t\right)\right\|^2, \tag{1}$$

where $\mathcal{U}(1,T)$ is the uniform distribution on $\{1,\ldots,T\}$, $\omega_t = (1-\alpha_t)^2/2\sigma_t^2\alpha_t\left(1-\bar{\alpha}_t\right)$ with $\bar{\alpha}_t = \prod_{s=0}^t \alpha_s$, $\sigma_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}(1-\alpha_t)$, and $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$. During inference, the learned model $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t\right)$ generates a sample $\mathbf{x}_0$ by gradually denoising the random noise $\mathbf{x}_T \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$ according to $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t,t\right)\right) + \sigma_t\mathbf{z}$, where $\mathbf{z} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$. The DPM suffers from slow sampling, as thousands of denoising steps are usually required to generate samples (Ho et al., 2020; Yang et al., 2023). To improve efficiency, the denoising diffusion implicit model (DDIM) (Song, Meng, & Ermon, 2021) uses a non-Markov diffusion process to achieve 10× faster sampling. DPM-Solver (Lu, Zhou, Bao, Chen, Li, & Zhu, 2022) formulates DPM as diffusion ordinary differential equations and reduces the number of denoising steps from 1000 to about 10. DPM-Solver++ (Lu et al., 2023) further improves the robustness of classifier guidance in DPM-Solver by using a high-order solver.

The DPM is unable to control the class labels of generated samples. To handle this issue, conditional diffusion probabilistic model (CDPM) (Nichol & Dhariwal, 2021) incorporates label information into the model by adding a label embedding to the time embedding. Dhariwal and Nichol (2021) further improve the CDPM by embedding a condition (i.e., class label) into the AdaGN layers (Dhariwal & Nichol, 2021) of the UNet (Ronneberger, Fischer, & Brox, 2015). Moreover, classifier guidance (Dhariwal & Nichol, 2021) is also introduced, and the denoising process of class c is modified as $\tilde{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, c) = \boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t, c) - s\sigma_t\nabla_{\mathbf{x}_t}\log p_{\boldsymbol{\phi}}(c|\mathbf{x}_t, t)$, where $\boldsymbol{\phi}$ is a learned classifier and s is a scale factor. Diffusion models can learn complex data distributions from limited samples (Dhariwal & Nichol, 2021) and generate samples. To this end, this paper proposes DCDM to learn the distribution of labeled target samples by domain guidance and class condition.

3. Domain-guided conditional diffusion model

In UDA, we are given a labeled source domain $D_s = \{(\mathbf{x}^s, y^s)\}$ and an unlabeled target domain $D_\tau = \{\mathbf{x}^\tau\}$. Let $N_s = |D_s|$ and $N_\tau = |D_\tau|$ be the number of samples in D_s and D_τ, respectively. Usually, N_τ is much smaller than N_s (Yang, Zhang, Dai, & Pan, 2020). UDA aims to train a model from $D_s \cup D_\tau$, and then uses this model to make predictions on D_τ. However, the target domain has limited samples, which are also unlabeled, making transfer more difficult.

Previous augmentation-based methods adopt the image-to-image translation strategy (Hoffman et al., 2018) or construct intermediate domains to bridge the source and target domains (Xia, Jing, & Ding, 2023; Yang, Xia, et al., 2020), rather than generate labeled samples in the target domain. Therefore, these methods fail to address the data scarcity problem of UDA. In this paper, we propose the Domain-guided Conditional Diffusion Model (DCDM), which generates "labeled" target domain samples. Moreover, unlike existing conditional GANs (Mirza & Osindero, 2014; Odena, Olah, & Shlens, 2017) and variational autoencoders (VAEs) (Sohn, Lee, & Yan, 2015), the class label and domain information are decoupled in DCDM to facilitate label information sharing between the source and target domains. For each generated sample, they are separately controlled by using the class condition and domain guidance, respectively. As illustrated in Fig. 1, the DCDM has two stages: *label-conditioned training* and *domain-guided generation*. In label-conditioned training, we train a label-conditioned diffusion model on the source and target domain samples to control the labels of generated samples. In domain-guided generation, we first train a domain classifier, which is then used in domain-guidance sampling to generate samples for the target domain. Finally, the generated samples are combined with the source samples as an augmented source domain to train a UDA model. The augmented source domain can reduce domain shifts, benefiting existing UDA methods.

3.1. Label-conditioned training

To generate more target domain samples, a straightforward solution is to train a DPM on the target domain. Analogous to (1), a noise prediction model $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t^\tau, t\right)$ can be trained by minimizing the following loss as

$$\mathbb{E}_{t\sim\mathcal{U}(1,T),\mathbf{x}_0^\tau\sim\mathcal{T},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\omega_t\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t^\tau, t\right)\right\|^2. \tag{2}$$

However, this cannot control the labels of the samples and cannot ensure that the generated samples cover all classes.

To alleviate this problem, one can use the label embedding of $\mathbf{x}_0^\tau$ as condition $\mathbf{c}$ and feed this to a CDPM (Nichol & Dhariwal, 2021) $\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t^\tau, t, \mathbf{c}\right)$ (Section 2.2). However, labels for the target domain samples are not available. To address this issue, we first train a UDA model $f^\star$ by minimizing the commonly-used objective as

$$f^\star = \arg\min_f \frac{1}{N_s}\sum_{(\mathbf{x},y)\in D_s}\ell(f(\mathbf{x}), y) + \beta\mathcal{R}(D_s, D_\tau), \tag{3}$$

where $\ell(\cdot,\cdot)$ is a supervised loss (e.g., cross-entropy loss for classification problems), $\beta > 0$ is a tradeoff parameter, and $\mathcal{R}(\cdot,\cdot)$ is a loss used to bridge the source and target domains (e.g., domain discrepancy loss in discrepancy-based methods Jin et al., 2020; Long et al., 2015; Zhang et al., 2019, or domain discrimination loss in adversarial-based methods Ganin et al., 2016; Rangwani et al., 2022; Zhang et al., 2023). We then use $f^\star$'s predictions (denoted by $\hat{\mathbf{y}}$) on the target domain samples as their pseudo-labels. The loss in problem (2) is consequently changed to

$$\mathbb{E}_{t\sim\mathcal{U}(1,T),\mathbf{x}_0^\tau\sim\mathcal{T},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\omega_t\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t^\tau, t, f^\star(\mathbf{x}_0^\tau)\right)\right\|^2. \tag{4}$$

Using more training data is beneficial to DPM training. It can also improve the fidelity of the generated samples (Croitoru, Hondru, Ionescu, & Shah, 2023). Thus, instead of using only the target domain samples to train $\boldsymbol{\epsilon}_\theta$ as in problem (4), we use both the source and target domain samples to minimize the following loss

$$\mathbb{E}_{t\sim\mathcal{U}(1,T),\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\omega_t\left(\mathbb{E}_{(\mathbf{x}_0,\cdot)\sim S}\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t, \mathbf{c}\right)\right\|^2 + \mathbb{E}_{\mathbf{x}_0\sim\mathcal{T}}\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t, \mathbf{c}\right)\right\|^2\right), \tag{5}$$

where $\mathbf{c}$ is the (true) label when $\mathbf{x}_0$ is from the source domain, and the pseudo-label predicted by $f^\star$ when $\mathbf{x}_0$ is from the target domain. The ground truth labels of source samples can reduce the noise brought by pseudo-labels. The whole label-conditioned training procedure is shown in Algorithm 1. After label-conditioned training, we obtain the optimized diffusion model $\boldsymbol{\epsilon}_\theta^\star$ and can generate images using this model. Specifically, to generate a sample $\mathbf{x}_0$ from class y, we gradually denoise the random noise sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ according to $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t, y\right)\right) + \sigma_t\mathbf{z}$, where $\mathbf{z}\sim\mathcal{N}(\mathbf{0}, \mathbf{I})$.

3.2. Domain-guided generation

While label-conditioned training can control the generated sample's class label, it does not control its domain. In UDA, as we focus on the performance in the target domain, it is more crucial to generate samples for the target domain.

To achieve this, inspired by classifier guidance (Dhariwal & Nichol, 2021), we adopt a domain classifier to guide the generation towards the target domain. Specifically, we first train a domain classifier (with parameter $\boldsymbol{\phi}$) on the noisy images $\mathbf{x}_t$'s by minimizing the following loss as

$$\boldsymbol{\phi}^\star = \arg\min_{\boldsymbol{\phi}}\sum_{t=1}^{T}\sum_{(\mathbf{x}_0,\cdot)\in D_s}\ell(\boldsymbol{\phi}(\mathbf{x}_t^s, t), d^s) + \sum_{\mathbf{x}_0\in D_\tau}\ell(\boldsymbol{\phi}(\mathbf{x}_t^\tau, t), d^\tau), \tag{6}$$

where $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$, $\ell(\cdot,\cdot)$ is the cross-entropy loss, d^s and d^τ are the ground-truth source and target domain labels of $\mathbf{x}_0$, respectively. After obtaining $\boldsymbol{\phi}^\star$, the gradient $\nabla_{\mathbf{x}_t}\log p_{\boldsymbol{\phi}^\star}(d^\tau|\mathbf{x}_t, t)$, where $p_{\boldsymbol{\phi}^\star}(d^\tau|\mathbf{x}_t, t)$ is the predicted domain probability, is used to guide the sampling process towards the target domain. Then, the domain-guided noise prediction model becomes

$$\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t, \hat{\mathbf{c}}) = \boldsymbol{\epsilon}_{\theta^\star}(\mathbf{x}_t, t, y) - s\sqrt{1-\bar{\alpha}_t}\,\nabla_{\mathbf{x}_t}\log p_{\boldsymbol{\phi}^\star}(d^\tau|\mathbf{x}_t, t), \tag{7}$$

where s is a scale factor, and $\hat{\mathbf{c}} = [y, d^\tau]$ with y being the class label of images we aim to generate.

To accelerate the sampling procedure, we use DPM-Solver++ (Lu et al., 2023) to generate samples D_g for the target domain. Given an initial noisy $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and time steps $\{t_i\}_{i=0}^M$, this multi-step second-order solver iterates as

$$\begin{aligned}\mathbf{x}_{t_i} = {}& \sqrt{\alpha_{t_i}}\left(e^{-b_i}-1\right)\left(\frac{b_i}{2b_{i-1}}\mathbf{x}_\theta\left(\mathbf{x}_{t_{i-2}}, t_{i-2}, \hat{\mathbf{c}}\right)\right.\\ &\left.-\left(1+\frac{b_i}{2b_{i-1}}\right)\mathbf{x}_\theta\left(\mathbf{x}_{t_{i-1}}, t_{i-1}, \hat{\mathbf{c}}\right)\right) + \frac{\sqrt{1-\bar{\alpha}_{t_i}}}{\sqrt{1-\bar{\alpha}_{t_{i-1}}}}\mathbf{x}_{t_{i-1}},\end{aligned} \tag{8}$$

for $i = 0, \ldots, M$, where $\mathbf{x}_\theta(\mathbf{x}_t, t, \hat{\mathbf{c}}) = \frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\tilde{\boldsymbol{\epsilon}}_\theta(\mathbf{x}_t, t, \hat{\mathbf{c}})}{\sqrt{\bar{\alpha}_t}}$, $\lambda_t = \frac{1}{2}\log\left(\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}\right)$, and $b_i = \lambda_{t_i} - \lambda_{t_{i-1}}$. The whole domain-guided generation procedure is shown in Algorithm 2.

To reduce the domain shift, we combine the samples D_g generated from Algorithm 2 with the original source domain samples D_s to form an augmented source domain data $D_{\hat{s}}$. The effectiveness of this combination will be verified in Section 4.4. A UDA model is then used to transfer from $D_{\hat{s}}$ to the target domain D_τ by minimizing the following objective, which is analogous to that in (3):

$$\hat{f} = \arg\min_f \frac{1}{N_{\hat{s}}}\sum_{(\mathbf{x},y)\in D_{\hat{s}}}\ell(f(\mathbf{x}), y) + \beta\mathcal{R}(D_{\hat{s}}, D_\tau), \tag{9}$$

where $N_{\hat{s}} = |D_{\hat{s}}| = N_s + N_g$ denotes the number of samples in $D_{\hat{s}}$. Finally, the learned $\hat{f}$ is evaluated on the target samples.

The whole DCDM algorithm is shown in Algorithm 3. Note that DCDM can be integrated into any UDA method (i.e., step 1 in Algorithm 1 and step 9 in Algorithm 3). For the experiments in Section 4, DCDM is combined with MCC (Jin et al., 2020), ELS (Zhang et al., 2023), and SSRT (Sun et al., 2022).

Algorithm 1 Label-conditioned training.

Input: Source domain D_s, target domain D_τ, step size γ, decreasing sequence $(\alpha_1, \ldots, \alpha_T)$, mini-batch B;

1: Train a UDA model $f^\star$ on $D_s \cup D_\tau$;
2: **repeat**
3: Sample a mini-batch of images $\{\mathbf{x}_{0,\mathrm{i}}\}_{i=1}^B$ from $D_s \cup D_\tau$;
4: **for** $i = 1, \ldots, B$ **do**
5: **if** $\mathbf{x}_{0,i} \in D_\tau$ **then**
6: $\mathbf{c}_i = f^\star(\mathbf{x}_{0,i})$;
7: **else**
8: $\mathbf{c}_i = y_i$;
9: **end if**
10: **end for**
11: $t \sim \mathcal{U}(1, T)$;
12: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
13: $\bar{\alpha}_t = \prod_{j=1}^t \alpha_j$, $\sigma_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}(1-\alpha_t)$;
14: $\mathbf{x}_{t,i} = \sqrt{\bar{\alpha}_t}\mathbf{x}_{0,i} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$; $i = 1, \ldots, B$;
15: $\omega_t = \frac{(1-\alpha_t)^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}$;
16: $\mathcal{L}_{\text{mini-batch}}(\boldsymbol{\theta}) = \frac{\omega_t}{B}\sum_{i=1}^B\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_{t,i}, t, \mathbf{c}_i\right)\right\|^2$;
17: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\text{mini-batch}}(\boldsymbol{\theta})$;
18: **until** converged.
19: **return** $\boldsymbol{\theta}^\star$.

3.3. Analysis

In this section, we provide analyses for the proposed DCDM method.

Algorithm 2 Domain-guided generation.

Input: Initial $\mathbf{x}_T$, time steps $\{t_i\}_{i=0}^{M}$, the optimized noise prediction model $\epsilon_{\theta^\star}$ obtained from Algorithm 1, domain classifier model $\boldsymbol{\phi}^\star$, target domain label d^τ, and guidance scale s, decreasing sequence $\alpha_1, \dots, \alpha_T$;

$\mathbf{x}_{t_0} = \mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$;
Sample a class y;
$\hat{\mathbf{c}} = [y, d^\tau]$;
for $i = 1, \dots, M$ **do**
  $\bar{\alpha}_{t_i} = \prod_{j=1}^{t_i} \alpha_j$;
  $\lambda_{t_i} = \frac{1}{2} \log\left(\frac{\bar{\alpha}_{t_i}}{1-\bar{\alpha}_{t_i}}\right)$;
  $b_i = \lambda_{t_i} - \lambda_{t_{i-1}}$;
  Compute guidance $\mathbf{d}_{t_i} = \sqrt{1-\bar{\alpha}_{t_i}} \nabla_{\mathbf{x}_{t_i}} \log p_{\boldsymbol{\phi}^\star}(d|\mathbf{x}_{t_i}, t_i)$;
  $\tilde{\epsilon}_\theta(\mathbf{x}_{t_i}, t_i, \hat{\mathbf{c}}) = \epsilon_{\theta^\star}(\mathbf{x}_{t_i}, t_i, y) - s\mathbf{d}_{t_i}$;
  if $i = 1$ **then**
    $\mathbf{x}_{t_i} = \frac{\sqrt{1-\bar{\alpha}_{t_i}}}{\sqrt{1-\bar{\alpha}_{t_{i-1}}}} \mathbf{x}_{t_{i-1}} + \sqrt{\bar{\alpha}_{t_i}}(1-e^{-b_i})\left(\mathbf{x}_{t_{i-1}} - \frac{\sqrt{1-\bar{\alpha}_{t_i}}}{\sqrt{\bar{\alpha}_{t_i}}} \tilde{\epsilon}_\theta(\mathbf{x}_{t_{i-1}}, t_{i-1}, \hat{\mathbf{c}})\right)$;
  else
    Compute $\mathbf{x}_{t_i}$ according to Eq. (8);
  end if
end for
return $(\mathbf{x}_{t_M}, y)$.

Algorithm 3 UDA via Domain-guided Conditional Diffusion Model (DCDM).

Input: Source domain $\mathcal{D}_s$, target domain $\mathcal{D}_\tau$, number of generated samples N_g;

Train a label-conditioned diffusion model $\epsilon_{\theta^\star}$ on $\mathcal{D}_s \cup \mathcal{D}_\tau$ by Algorithm 1;
Train a domain classifier $\boldsymbol{\phi}^\star$ on $\mathcal{D}_s \cup \mathcal{D}_\tau$;
$\mathcal{D}_g = \emptyset$;
for $i = 1, \dots, N_g$ **do**
  Generate $(\mathbf{x}_i, y_i)$ with $\boldsymbol{\phi}^\star$ by Algorithm 2;
  $\mathcal{D}_g = \mathcal{D}_g \cup \{(\mathbf{x}_i, y_i)\}$;
end for
$\mathcal{D}_{\hat{s}} = \mathcal{D}_g \cup \mathcal{D}_s$;
Train a UDA model $\hat{f}$ on $\mathcal{D}_{\hat{s}} \cup \mathcal{D}_\tau$;
Evaluate the learned $\hat{f}$ on $\mathcal{D}_\tau$;

3.3.1. Connection to score matching

The proposed DCDM learns the distribution of labeled target samples by domain guidance and class condition. In the following, we show how the domain guidance and class condition relate to the score matching (Song, Durkan, Murray & Ermon, 2021), which formulates diffusion models from a different perspective.

Let y be the class label and d^τ be the target domain label. With an assumption that y and d^τ are independent, we can obtain the distribution of the labeled target domain sample as

$$p(\mathbf{x} \mid y, d^\tau) = \frac{p(\mathbf{x})p(y, d^\tau|\mathbf{x})}{p(y, d^\tau)} = \frac{p(\mathbf{x})p(y \mid \mathbf{x})p(d^\tau \mid \mathbf{x})}{p(y)p(d^\tau)} = p(\mathbf{x} \mid y)\frac{p(d^\tau \mid \mathbf{x})}{p(d^\tau)},$$

Thus, the score function can be represented as

$$\nabla_{\mathbf{x}} \log p(\mathbf{x} \mid y, d^\tau) = \nabla_{\mathbf{x}} \log p(\mathbf{x} \mid y) + \nabla_{\mathbf{x}} \log p(d^\tau \mid \mathbf{x}).$$

In the proposed DCDM method, $\nabla_{\mathbf{x}} \log p(\mathbf{x} \mid y)$ is learned by the conditional diffusion model $\epsilon_{\theta^\star}$, and $\nabla_{\mathbf{x}} \log p(d^\tau \mid \mathbf{x})$ is learned by the domain classifier $\boldsymbol{\phi}^\star$. Accordingly, we can generate "labeled" target domain samples by $\nabla_{\mathbf{x}} \log p(\mathbf{x} \mid y, d^\tau)$ to facilitate domain adaptation.

3.3.2. Generalization properties of DCDM

In this section, we study the generalization properties of DCDM. Let $\mathcal{E}_s(f, f') = P_{(\mathbf{x},y)\sim\mathcal{S}}(f(\mathbf{x}) \neq f'(\mathbf{x}))$, $\mathcal{E}_\tau(f, f') = P_{(\mathbf{x},y)\sim\mathcal{T}}(f(\mathbf{x}) \neq f'(\mathbf{x}))$, $\mathcal{E}_g(f, f') = P_{(\mathbf{x},y)\sim\mathcal{G}}(f(\mathbf{x}) \neq f'(\mathbf{x}))$, and $\mathcal{E}_{\hat{s}}(f, f') = P_{(\mathbf{x},y)\sim\hat{\mathcal{S}}}(f(\mathbf{x}) \neq f'(\mathbf{x}))$ be the disagreements between models f and f' on data distributions $\mathcal{S}$, $\mathcal{T}$, $\mathcal{G}$, and $\hat{\mathcal{S}}$, respectively, where $\mathcal{S}$ and $\mathcal{T}$ denote the source and target distributions, respectively, and $\mathcal{G}$ and $\hat{\mathcal{S}}$ denote the distributions of $\mathcal{D}_g$ and $\mathcal{D}_{\hat{s}}$, respectively. When f' is an oracle (i.e., $f'(\mathbf{x})$ is the ground-truth label y of $\mathbf{x}$), we simply write $\mathcal{E}_s(f) = P_{(\mathbf{x},y)\sim\mathcal{S}}(f(\mathbf{x}) \neq y)$, $\mathcal{E}_\tau(f) = P_{(\mathbf{x},y)\sim\mathcal{T}}(f(\mathbf{x}) \neq y)$, $\mathcal{E}_g(f) = P_{(\mathbf{x},y)\sim\mathcal{G}}(f(\mathbf{x}) \neq y)$, and $\mathcal{E}_{\hat{s}}(f) = P_{(\mathbf{x},y)\sim\hat{\mathcal{S}}}(f(\mathbf{x}) \neq y)$ (the expected risks of f on $\mathcal{S}$, $\mathcal{T}$, $\mathcal{G}$, and $\hat{\mathcal{S}}$ respectively). The corresponding empirical risks are $\hat{\mathcal{E}}_s(f) = \frac{1}{N_s}\sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}_s} \mathbb{I}(f(\mathbf{x}_i) \neq y_i)$, $\hat{\mathcal{E}}_\tau(f) = \frac{1}{N_\tau}\sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}_\tau} \mathbb{I}(f(\mathbf{x}_i) \neq y_i)$, $\hat{\mathcal{E}}_g(f) = \frac{1}{N_g}\sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}_g} \mathbb{I}(f(\mathbf{x}_i) \neq y_i)$, and $\hat{\mathcal{E}}_{\hat{s}}(f) = \frac{1}{N_{\hat{s}}}\sum_{(\mathbf{x}_i,y_i)\in\mathcal{D}_{\hat{s}}} \mathbb{I}(f(\mathbf{x}_i) \neq y_i)$, where $\mathbb{I}(\cdot)$ is the indicator function. Let $f_s^\star$ be the classifier trained from the source domain. We have the following Proposition.

Proposition 3.1. *Denote the hypothesis space for classifier $\hat{f}$ (in problem (9)) by $\mathcal{H}$, and its VC dimension by V. For $\delta > 0$, with probability $1 - 2\delta$, the expected risk $\mathcal{E}_\tau(\hat{f})$ is bounded as:*

$$\mathcal{E}_\tau(\hat{f}) \le \eta\left(\hat{\mathcal{E}}_s\left(f_s^\star\right) + \frac{1}{2}\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_\tau) + C\right) + \varepsilon(\delta, V, N_{\hat{s}}) + (1-\eta)\left(\hat{\mathcal{E}}_g(f_s^\star) + \frac{1}{2}\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_g, \mathcal{D}_\tau) + C\right), \tag{10}$$

where $\eta = \frac{N_s}{N_{\hat{s}}}$, $C = 4\sqrt{\frac{2V\log(2N_{\hat{s}}) + \log\frac{2}{\delta}}{N_{\hat{s}}}}$, $\varepsilon(\delta, V, N_{\hat{s}}) = \sqrt{\frac{1}{2N_{\hat{s}}}\ln\frac{2V}{\delta}}$, and $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}, \mathcal{D}')$ is the empirical $\mathcal{H}\Delta\mathcal{H}$-distance between $\mathcal{D}$ and $\mathcal{D}'$ (Ben-David et al., 2010).

Proof. Using Theorem 2 in Ben-David et al. (2010), for any $\hat{f}, \tilde{f} \in \mathcal{H}$, with probability $1 - \delta$, we have

$$\mathcal{E}_\tau(\hat{f}) \le \mathcal{E}_{\hat{s}}(\hat{f}) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{S}}, \mathcal{T}). \tag{11}$$

$d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{S}}, \mathcal{T})$, the $\mathcal{H}\Delta\mathcal{H}$-distance[4] between $\hat{\mathcal{S}}$ and $\mathcal{T}$, satisfies

$$\begin{aligned}
&\frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{S}}, \mathcal{T}) \\
&= \sup_{\hat{f},\tilde{f}} |\mathcal{E}_\tau(\hat{f}, \tilde{f}) - \mathcal{E}_{\hat{s}}(\hat{f}, \tilde{f})| \\
&= \sup_{\hat{f},\tilde{f}} (|\eta(\mathcal{E}_\tau(\hat{f}, \tilde{f}) - \mathcal{E}_s(\hat{f}, \tilde{f})) + (1-\eta)(\mathcal{E}_\tau(\hat{f}, \tilde{f}) - \mathcal{E}_g(\hat{f}, \tilde{f}))|) \\
&\le \frac{1}{2}\eta d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) + \frac{1}{2}(1-\eta)d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{G}, \mathcal{T}).
\end{aligned} \tag{12}$$

Combining inequalities (11) and (12) gives

$$\begin{aligned}
&\mathcal{E}_\tau(\hat{f}) \\
&\le \mathcal{E}_{\hat{s}}(\hat{f}) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{S}}, \mathcal{T}) \\
&= \mathcal{E}_{\hat{s}}(\hat{f}) + \hat{\mathcal{E}}_{\hat{s}}\left(f_s^\star\right) - \hat{\mathcal{E}}_{\hat{s}}\left(f_s^\star\right) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{S}}, \mathcal{T}) \\
&\le \hat{\mathcal{E}}_{\hat{s}}(\hat{f}) + \varepsilon(\delta, V, N_{\hat{s}}) + \hat{\mathcal{E}}_{\hat{s}}\left(f_s^\star\right) - \hat{\mathcal{E}}_{\hat{s}}\left(f_s^\star\right) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{S}}, \mathcal{T}) && (13) \\
&\le \hat{\mathcal{E}}_{\hat{s}}\left(f_s^\star\right) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{S}}, \mathcal{T}) + \varepsilon(\delta, V, N_{\hat{s}}) && (14) \\
&\le \eta\hat{\mathcal{E}}_s\left(f_s^\star\right) + (1-\eta)\hat{\mathcal{E}}_g\left(f_s^\star\right) + \frac{1}{2}\left[\eta d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) + (1-\eta)\, d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{G}, \mathcal{T})\right] \\
&\quad + \varepsilon(\delta, V, N_{\hat{s}}) && (15) \\
&= \eta\left(\hat{\mathcal{E}}_s\left(f_s^\star\right) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T})\right) + \varepsilon(\delta, V, N_{\hat{s}}) \\
&\quad + (1-\eta)\left(\hat{\mathcal{E}}_g(f_s^\star) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{G}, \mathcal{T})\right). \\
&\le \eta\left(\hat{\mathcal{E}}_s\left(f_s^\star\right) + \frac{1}{2}\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_s, \mathcal{D}_\tau) + C\right) + \varepsilon(\delta, V, N_{\hat{s}})
\end{aligned}$$

[4] For any $f, f' \in \mathcal{H}$, $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}, \mathcal{D}') = 2\sup_{f,f'\in\mathcal{H}} |P_{\mathbf{x}\sim\mathcal{D}}(f(\mathbf{x}) \neq f'(\mathbf{x})) - P_{\mathbf{x}\sim\mathcal{D}'}(f(\mathbf{x}) \neq f'(\mathbf{x}))|$

(a) *Office-31*. (b) *Office-Home*. (c) *VisDA-2017*. (d) *DomainNet*.

Fig. 2. Example images from *Office-31*, *Office–Home*, *VisDA-2017*, and *DomainNet* datasets.

$$+ (1-\eta)\left(\hat{\mathcal{E}}_g(f_s^\star) + \frac{1}{2}\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_g, \mathcal{D}_\tau) + C\right), \tag{16}$$

where inequality (13) holds with probability $1-\delta$ according to Eq. (2.1) in Abu-Mostafa, Magdon-Ismail, and Lin (2012), inequality (14) holds since $\hat{f}$ is optimized by the UDA methods based on the augmented source domain and target domain while $f_s^\star$ is optimized only in the source domain, leading to $\hat{\mathcal{E}}_{\hat{s}}(\hat{f}) \leq \hat{\mathcal{E}}_{\hat{s}}\left(f_s^\star\right)$, and inequality (15) holds due to $\hat{\mathcal{E}}_{\hat{s}}(f_s^\star) = \eta\hat{\mathcal{E}}_s\left(f_s^\star\right) + (1-\eta)\hat{\mathcal{E}}_g\left(f_s^\star\right)$ and inequality (12). Then we reach the conclusion. □

Proposition 3.1 shows that the expected risk of DCDM on the target domain is upper-bounded by a convex combination of two terms with weights η and $1-\eta$. The first one is determined by the empirical risk in the source domain and the distance between the source and target domains, while the second term depends on the empirical risk of the generated domain and its distance to the target domain.

4. Experiments

In this section, we empirically evaluate the proposed DCDM on a number of benchmark datasets.

4.1. Settings

Datasets. Experiments are performed on five UDA benchmark datasets that have been commonly used (Jin et al., 2020; Sun et al., 2022; Zhang et al., 2023). Example images are shown in Fig. 2, and dataset statistics are detailed in Table 1. (i) *Office-31* (Saenko, Kulis, Fritz, & Darrell, 2010) contains 4,110 images from 31 classes of three domains (i.e., Amazon (A), DSLR (D), and Webcam (W)). Six transfer tasks (A→W, D→W, W→D, A→D, D→A, W→A) are constructed. (ii) *Office–Home* (Venkateswara, Eusebio, Chakraborty, & Panchanathan, 2017) contains 15,500 images from 65 classes of four domains (i.e., Art (Ar), Clipart (Cl), Product (Pr), and Real-World (Rw)). All combinations of domain transfer are considered, leading to a total of 12 transfer tasks. (iii) *VisDA-2017* (Peng et al., 2017) is a large-scale synthetic-to-real dataset with 207,785 images from 12 classes of two domains (i.e., Synthetic and Real). Following Jin et al. (2020), Rangwani et al. (2022), we consider the transfer task Synthetic → Real. (iv) *miniDomainNet* (Zhou, Yang, Qiao, & Xiang, 2021) is a subset of DomainNet (Peng et al., 2019), with 140,006 images from 126 classes of four domains (i.e., Clipart (C), Painting (P), Real (R), and Sketch (S)). Following Xie, Li, Zhang, and Liu (2023), 12 transfer tasks are constructed. (v) *DomainNet* (Peng et al., 2019) is a large-scale dataset contains about 0.6 million images of 345 classes in six domains (i.e., Clipart (clp), Infograph (inf), Painting (pnt), Quickdraw (qdr), Real (rel), and Sketch (skt)).

Baselines. We compare with empirical risk minimization (ERM) (Vapnik, 1999) and a number of UDA methods, including (i) discrepancy-based methods: AFN (Xu et al., 2019), MDD (Zhang et al., 2019), MCC (Jin et al., 2020), (ii) adversarial-based methods: DANN (Ganin et al., 2016), CDAN (Long et al., 2018), SDAT (Rangwani et al., 2022), ELS (Zhang et al., 2023), TVT (Yang, Liu, et al., 2023), SSRT (Sun et al., 2022), GH++ (Huang et al., 2024), and (iii) augmentation-based methods: BDG (Yang, Xia, et al., 2020), MSGD (Xia et al., 2023), and Adapt Anything (Chen et al., 2023). We combine the proposed DCDM with the state-of-the-art UDA methods (namely, MCC, ELS, and SSRT). Note that we use the same UDA method in both the *label-conditioned training* and *domain-guided generation* steps, though in general they can be different.

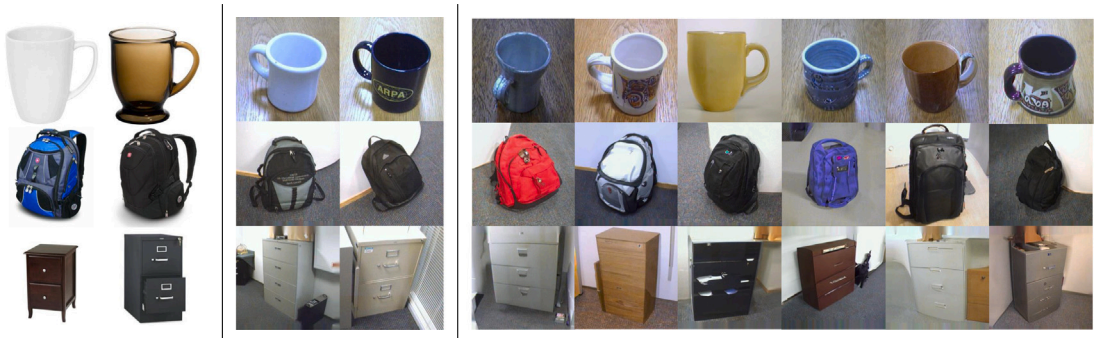
Table 1
Statistics of the datasets used.

Dataset	Number of images	Number of classes	Number of domains	Number of tasks
Office-31	4,110	31	3	6
Office–Home	15,500	65	4	12
VisDA-2017	207,785	12	2	1
miniDomainNet	140,006	126	4	12
DomainNet	586,575	345	6	30

Implementation Details. To initialize ϵ_θ, we use the CDPM in Dhariwal and Nichol (2021), which is pretrained on *ImageNet* and has a U-Net (Ronneberger et al., 2015) architecture. The resolutions of the generated images are 128 × 128 for the *miniDomainNet* dataset, and 256 × 256 for the other datasets. The images in the source and target domains are processed to the same resolution as the generated images. For each class, 200, 200, 2,000, 200, and 100 images are generated for the *Office-31*, *Office–Home*, *VisDA-2017*, *miniDomainNet*, and *DomainNet* datasets, respectively. TVT, SSRT, and GH++ adopt the ViT-base backbone (Dosovitskiy, 2020) for all datasets. Apart from these methods, following Rangwani et al. (2022), Zhou et al. (2021), the *ResNet-50* (He, Zhang, Ren, & Sun, 2016) is used as the backbone for the UDA model on *Office-31* and *Office–Home*, while the *ResNet-101* is used for *VisDA-2017*, and *ResNet-18* for *miniDomainNet*. The learning rate scheduler follows Ganin et al. (2016). For ELS+DCDM, the initial learning rate is 0.002 for *Office-31* and *VisDA-2017*, and 0.01 for the other two datasets. For MCC+DCDM, the initial learning rate is 0.002 for *VisDA-2017*, and 0.005 for the other three datasets. For SSRT+DCDM, the learning rate follows SSRT (Sun et al., 2022). All experiments are run on an NVIDIA V100 GPU.

4.2. Results

Tables 2–6 show the transfer accuracies on *Office-31*, *Office–Home*, *VisDA-2017*, *miniDomainNet*, and *DomainNet* datasets, respectively. According to the results on *Office-31*, we can see that SSRT+DCDM has the best performance on all tasks. Among them, on the three challenging tasks (i.e., A→W, W→A, and D→A), DCDM brings significant performance improvements to MCC, ELS, and SSRT. And DCDM performs better than the augmentation-based methods MSGD, BDG, and Adapt Anything. Moreover, SSRT+DCDM achieves the highest average accuracy on *Office-31*, *Office–Home*, *VisDA-2017*, and *DomainNet*, showing the proposed DCDM method is effective. Specifically, SSRT+DCDM is the best on nine tasks, seven categories, and 27 tasks of *Office–Home*, *VisDA-2017*, and *DomainNet*, respectively. On *Office–Home*, the proposed method achieves an average performance improvement of



(a) Source domain samples. (b) Target domain samples (c) Generated target domain samples.

Fig. 3. Real and generated images for the transfer task A→W from the *Office-31* dataset.

Fig. 4. Real source images (top) and generated source images (bottom) for the transfer task A→W from the *Office-31* dataset.

Table 2
Transfer accuracies (%) on *Office-31*. * denotes the ViT-base backbone. The best is in bold.

	A→W	D→W	W→D	A→D	D→A	W→A	Average
ERM (Vapnik, 1999)	77.07	96.60	99.20	81.08	64.11	64.01	80.35
DANN (Ganin et al., 2016)	89.85	97.95	99.90	83.26	73.28	73.75	86.33
AFN (Xu et al., 2019)	91.82	98.77	**100.00**	95.12	72.43	70.71	88.14
CDAN (Long et al., 2018)	92.42	98.62	**100.00**	91.44	74.61	72.80	88.32
MDD (Zhang et al., 2019)	93.55	98.66	**100.00**	93.92	75.29	73.95	89.23
SDAT (Rangwani et al., 2022)	91.32	98.83	**100.00**	95.25	76.97	73.19	89.26
BDG (Yang, Xia, et al., 2020)	93.60	99.00	**100.00**	93.60	73.20	72.00	88.50
MSGD (Xia et al., 2023)	95.50	99.20	**100.00**	95.60	77.30	77.00	90.80
Adapt Anything (Chen et al., 2023)	94.20	94.20	95.20	95.20	82.10	82.10	90.50
TVT* (Yang, Liu, et al., 2023)	96.35	99.37	**100.00**	96.39	84.91	86.05	93.85
SSRT+GH++* (Huang et al., 2024)	98.90	99.30	**100.00**	98.80	84.60	83.30	94.10
MCC (Jin et al., 2020)	94.09	98.32	99.67	94.25	75.89	75.46	89.61
MCC+DCDM	95.51	98.58	99.93	95.31	78.26	78.43	91.01
ELS (Zhang et al., 2023)	93.84	98.78	**100.00**	95.78	77.72	75.13	90.21
ELS+DCDM	96.90	98.91	**100.00**	97.46	79.79	77.74	91.80
SSRT* (Sun et al., 2022)	97.70	99.20	**100.00**	98.60	83.50	82.20	93.50
SSRT+DCDM*	**100.00**	**100.00**	**100.00**	**99.90**	**85.06**	**86.69**	**95.29**

1.47%, 1.80%, and 1.11% for MCC, ELS, and SSRT, respectively. On *VisDA*, DCDM achieves an average performance improvement of 3.24%, 2.89%, and 1.02% for MCC, ELS, and SSRT, respectively. On *miniDomainNet*, DCDM achieves an average performance improvement of 1.63% and 3.51% for MCC and ELS, respectively. On *DomainNet*, DCDM brings a noticeable average performance improvement (> +3%) to SSRT, verifying the usefulness of DCDM on this large-scale dataset.

4.3. Analysis of generated samples

Fig. 3 shows the generated target samples of the transfer task A→W on *Office-31*. As can be seen, the generated samples (in Fig. 3(c)) are of high quality and have a similar style to the real target domain samples (in Fig. 3(b)). For example, in the second row of Fig. 3, the real target domain has only black backpacks, while the generated images contain backpacks in various colors. One possible reason is that DCDM incorporates source domain samples during the diffusion training process, allowing it to generate backpacks with more colors related to the source domain (e.g., red and blue) while preserving the style of the target domain. Similarly, in the third row of Fig. 3, the target domain samples contain only metal-style file cabinets, while the generated images have cabinets of more styles (e.g., brown and black) that are related to the source domain samples.

(a) Source domain samples (b) Generated target domain samples

Fig. 5. Real source domain images and generated target domain images for the transfer task W→A from the *Office-31* dataset.

With domain guidance, the trained DCDM can also generate source domain images. Fig. 4 shows the real and generated source domain images for the transfer task A→W on *Office-31*. As can be seen, their styles are similar.

Traditional UDA methods may suffer from spurious correlation in the source domain (Bao, Chang, & Barzilay, 2022). For example, in the transfer task W→A from *Office-31*, all backpacks in the source domain W are black (Fig. 5(a)), while backpacks in the target domain are of various colors. However, since the target domain backpacks are unlabeled, they cannot be used directly. The UDA model may then mistakenly assume that all backpacks are black, and misclassify backpacks of the other colors in the target domain. To alleviate this issue, the proposed DCDM generates "labeled" target domain images in various colors, as shown in Fig. 5(b). By combining the generated data and source domain data, ELS+DCDM achieves an accuracy of 97.83% on the class 'backpacks', better than that of ELS (96.74%).

Table 3
Transfer accuracies (%) on *Office–Home*. * denotes the ViT-base backbone. The best is in bold.

	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Average
ERM (Vapnik, 1999)	44.06	67.12	74.26	53.26	61.96	64.54	51.91	38.90	72.94	64.51	43.84	75.39	59.39
DANN (Ganin et al., 2016)	52.53	62.57	73.20	56.89	67.02	68.34	58.37	54.14	78.31	70.78	60.76	80.57	65.29
AFN (Xu et al., 2019)	52.58	72.42	76.96	64.90	71.14	72.91	64.08	51.29	77.83	72.21	57.46	82.09	67.99
CDAN (Long et al., 2018)	54.21	72.18	78.29	61.97	71.43	72.39	62.96	55.68	80.68	74.71	61.22	83.68	69.12
MDD (Zhang et al., 2019)	56.37	75.53	79.17	62.95	73.21	73.55	62.56	54.86	79.49	73.84	61.45	84.06	69.75
SDAT (Rangwani et al., 2022)	58.20	77.46	81.35	66.06	76.45	76.41	63.70	56.69	82.49	76.02	62.09	85.24	71.85
BDG (Yang, Xia, et al., 2020)	51.50	73.40	78.70	65.30	71.50	73.70	65.10	49.70	81.10	74.60	55.10	84.80	68.70
MSGD (Xia et al., 2023)	58.70	76.90	78.90	70.10	76.20	76.60	69.00	57.20	82.30	74.90	62.70	84.50	72.40
Adapt Anything (Chen et al., 2023)	71.60	88.40	89.00	84.20	88.40	89.00	84.20	71.60	89.00	84.20	71.60	88.40	83.30
TVT* (Yang, Liu, et al., 2023)	74.89	86.82	89.47	82.78	87.95	88.27	79.81	71.94	90.13	85.46	74.62	90.56	83.56
SSRT+GH++* (Huang et al., 2024)	75.50	89.60	91.40	86.10	89.40	90.30	**85.40**	75.20	91.30	**86.90**	**79.70**	92.40	86.10
MCC (Jin et al., 2020)	56.83	79.81	82.66	67.80	77.02	77.82	66.98	55.43	81.79	73.95	61.41	85.44	72.24
MCC+DCDM	58.23	80.33	82.91	70.14	79.15	81.36	68.49	57.75	83.44	74.18	63.81	85.67	73.71
ELS (Zhang et al., 2023)	57.79	77.65	81.62	66.59	76.74	76.43	62.69	56.69	82.12	75.63	62.85	85.35	71.84
ELS+DCDM	60.35	78.81	82.74	69.59	80.53	79.55	65.16	58.26	83.11	75.81	64.18	85.55	73.64
SSRT* (Sun et al., 2022)	75.17	88.98	91.09	85.13	88.29	89.95	85.04	74.23	91.26	85.70	78.58	91.78	85.43
SSRT+DCDM*	**79.34**	**91.55**	**91.65**	**86.53**	**90.04**	**90.38**	84.96	**76.59**	**91.39**	84.47	78.92	**92.70**	**86.54**

Table 4
Transfer accuracies (%) on *VisDA-2017*. * denotes the ViT-base backbone. The best is in bold.

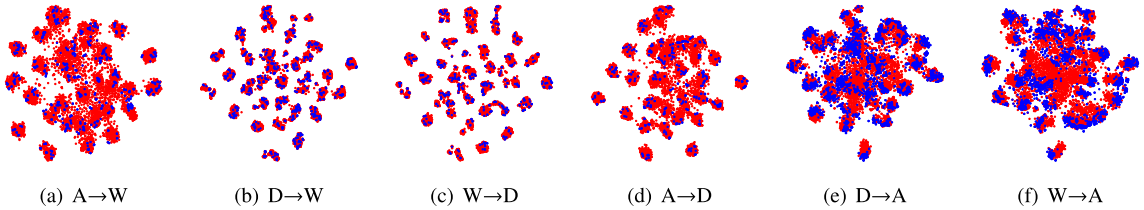
	Aero	Bicycle	Bus	Car	Horse	Knife	Motor	Person	Plant	Skate	Train	Truck	Mean
ERM (Vapnik, 1999)	81.71	22.46	54.08	76.21	74.83	10.69	83.81	18.71	80.88	28.66	79.66	5.98	51.47
DANN (Ganin et al., 2016)	94.75	73.47	83.46	47.91	87.00	88.30	88.47	77.18	88.16	90.05	87.21	42.26	79.02
AFN (Xu et al., 2019)	93.13	54.76	81.03	69.74	92.36	75.88	92.11	73.83	93.16	55.55	90.48	23.63	74.64
CDAN (Long et al., 2018)	94.55	74.41	82.22	58.92	90.56	96.22	89.71	78.90	86.11	89.06	84.81	43.42	80.74
MDD (Zhang et al., 2019)	92.68	65.26	82.29	66.78	91.68	92.09	93.18	79.67	92.12	84.95	83.85	48.66	81.10
SDAT (Rangwani et al., 2022)	94.51	83.56	74.28	65.78	93.00	95.83	89.61	80.04	90.86	91.47	84.95	54.93	83.23
Adapt Anything (Chen et al., 2023)	99.00	88.00	89.40	**92.10**	98.40	97.70	96.80	76.60	98.00	93.20	94.70	52.40	89.70
TVT* (Yang, Liu, et al., 2023)	92.92	85.58	77.51	60.48	93.60	98.17	89.35	76.40	93.56	92.02	91.69	55.73	83.92
SSRT+GH++* (Huang et al., 2024)	99.23	88.53	89.57	85.34	98.23	98.80	**96.95**	**83.68**	**95.37**	97.41	94.18	45.07	89.36
MCC (Jin et al., 2020)	95.26	86.14	77.12	69.98	92.83	94.84	86.52	77.78	90.26	90.98	85.68	52.52	83.32
MCC+DCDM	96.43	87.28	83.16	74.37	94.59	96.13	88.60	81.90	92.98	94.45	87.26	**61.56**	86.56
ELS (Zhang et al., 2023)	94.76	83.38	75.44	66.45	93.16	95.14	89.09	80.13	90.77	91.06	84.09	57.36	83.40
ELS+DCDM	96.20	84.79	83.15	73.28	94.76	96.58	90.99	82.21	92.98	93.37	87.49	59.70	86.29
SSRT* (Sun et al., 2022)	98.93	87.60	89.10	84.77	98.34	98.70	96.27	81.08	94.86	97.90	94.50	43.13	88.76
SSRT+DCDM*	**99.29**	**91.54**	**90.02**	79.03	**98.70**	**99.08**	95.77	81.98	94.99	**98.38**	**96.36**	52.18	**89.78**

Table 5
Transfer accuracies (%) on *miniDomainNet*. The best is in bold.

	C→P	C→R	C→S	P→C	P→R	P→S	R→C	R→P	R→S	S→C	S→P	S→R	Average
ERM (Vapnik, 1999)	39.48	53.27	42.93	49.55	68.18	41.99	49.30	55.52	37.35	54.60	45.33	53.08	49.22
DANN (Ganin et al., 2016)	45.94	56.12	49.40	50.72	65.61	50.07	55.15	60.55	49.95	58.54	54.64	58.99	54.64
AFN (Xu et al., 2019)	49.23	60.11	51.11	55.60	70.59	51.78	55.84	60.41	47.46	60.69	56.36	62.28	56.79
CDAN (Long et al., 2018)	47.99	58.50	51.17	56.36	68.71	53.01	61.15	62.85	53.44	60.89	55.90	60.88	57.57
MDD (Zhang et al., 2019)	48.53	61.75	52.32	59.74	70.62	55.43	62.18	62.22	54.04	63.07	58.55	64.50	59.41
SDAT (Rangwani et al., 2022)	50.97	62.42	53.91	60.57	69.97	55.85	64.39	64.83	55.86	64.07	59.43	64.28	60.55
MCC (Jin et al., 2020)	51.95	67.73	52.66	60.96	76.61	54.67	64.15	64.02	50.34	63.64	59.68	69.78	61.35
MCC+DCDM	55.28	**70.21**	54.42	63.25	**77.05**	55.74	**65.16**	**65.01**	51.96	65.76	60.83	**71.09**	62.98
ELS (Zhang et al., 2023)	50.11	61.45	53.02	60.77	70.61	56.04	62.43	64.16	54.89	63.93	59.19	64.47	60.09
ELS+DCDM	**56.14**	68.52	**55.51**	**64.96**	73.44	**58.23**	64.67	64.36	**57.66**	**67.23**	**62.56**	69.90	**63.60**

Table 6
Transfer accuracies (%) on *DomainNet*. * denotes the ViT-base backbone.

ERM	clp	inf	pnt	qdr	rel	skt	Avg.	MCC	clp	inf	pnt	qdr	rel	skt	Avg.	MDD	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	18.0	36.2	12.1	53.6	42.6	32.5	clp	–	19.9	39.6	9.0	56.9	43.6	33.8	clp	–	20.5	40.7	6.2	52.5	42.1	32.4
inf	41.0	–	35.7	4.7	52.9	31.2	33.1	inf	37.2	–	38.1	3.0	54.8	26.6	31.9	inf	33.0	–	33.8	2.6	46.2	24.5	28.0
pnt	46.0	18.5	–	6.2	59.9	38.5	34.4	pnt	48.4	19.7	–	4.4	61.1	41.2	35.0	pnt	43.7	20.4	–	2.8	51.2	41.7	32.0
qdr	13.6	1.2	2.1	–	6.0	10.2	6.6	qdr	18.5	3.9	9.2	–	17.6	13.0	12.4	qdr	18.4	3.0	8.1	–	12.9	11.8	10.8
rel	51.8	21.8	50.3	7.4	–	38.0	33.9	rel	55.1	22.5	54.0	4.7	–	37.8	34.8	rel	52.8	21.6	47.8	4.2	–	41.2	33.5
skt	54.8	15.6	38.6	14.1	49.9	–	34.6	skt	60.0	18.7	47.3	10.3	57.9	–	38.8	skt	54.3	17.5	43.1	5.7	54.2	–	35.0
Avg.	42.0	15.0	32.6	8.9	44.5	32.1	29.2	Avg.	43.8	16.9	37.6	6.3	49.6	32.4	32.4	Avg.	40.4	16.6	34.7	4.3	43.4	32.3	28.6
SSRT+GH++*	clp	inf	pnt	qdr	rel	skt	Avg.	SSRT*	clp	inf	pnt	qdr	rel	skt	Avg.	SSRT+DCDM*	clp	inf	pnt	qdr	rel	skt	Avg.
clp	–	35.8	60.9	22.9	76.0	61.8	51.5	clp	–	33.8	60.2	19.4	75.8	59.8	49.8	clp	–	33.0	62.0	32.9	78.1	61.2	53.4
inf	56.0	–	55.2	11.4	69.1	49.8	48.3	inf	55.5	–	54.0	9.0	68.2	44.7	46.3	inf	58.5	–	58.7	17.1	75.7	52.2	52.4
pnt	62.3	33.3	–	10.7	71.6	56.4	46.9	pnt	61.7	28.5	–	8.4	71.4	55.2	45.0	pnt	64.2	30.7	–	19.2	76.9	57.6	49.7
qdr	43.5	13.7	23.3	–	38.7	35.3	30.9	qdr	42.5	8.8	24.2	–	37.6	33.6	29.3	qdr	43.3	10.4	23.1	–	27.6	35.4	27.9
rel	70.8	39.8	66.3	10.8	–	61.5	49.8	rel	69.9	37.1	66.0	10.1	–	58.9	48.4	rel	70.3	37.2	66.4	21.0	–	59.5	50.8
skt	71.7	34.8	62.6	23.4	73.4	–	53.2	skt	70.6	32.8	62.2	21.7	73.2	–	52.1	skt	70.9	33.1	64.8	31.5	77.5	–	55.5
Avg.	60.9	31.5	53.7	15.8	65.8	53.0	46.8	Avg.	60.0	28.2	53.3	13.7	65.3	50.4	45.2	Avg.	61.4	28.9	55.0	24.3	67.1	53.2	48.3

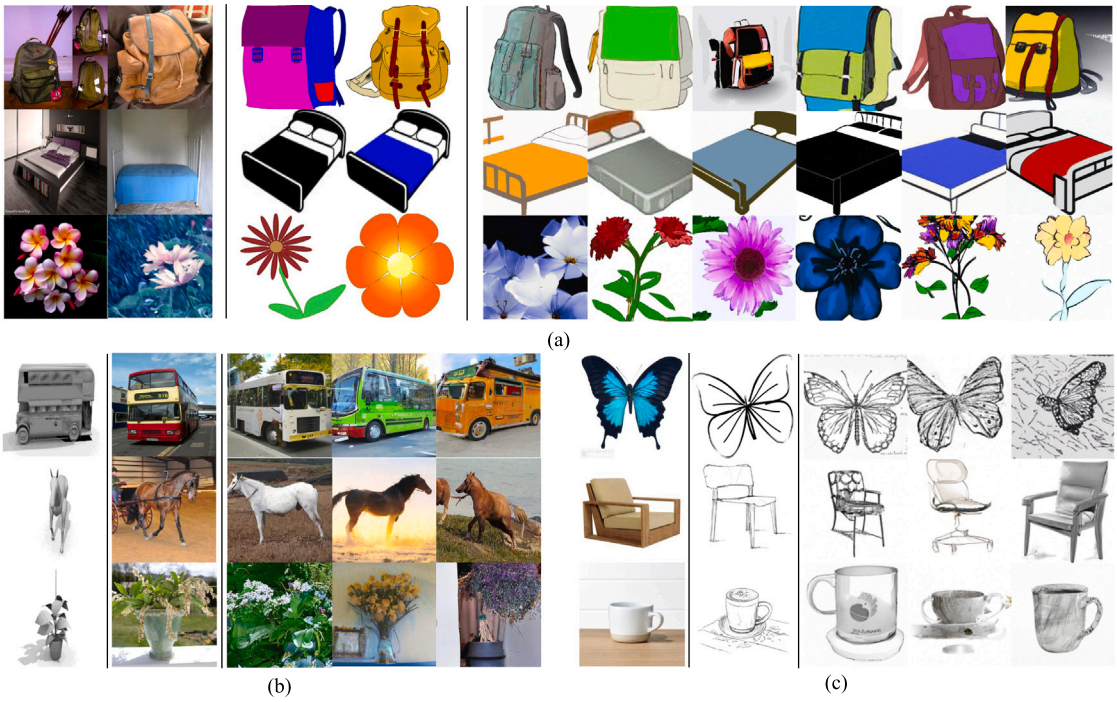


(a) A→W (b) D→W (c) W→D (d) A→D (e) D→A (f) W→A

Fig. 6. t-SNE visualization of the six tasks on *Office-31*. The generated and original target domain samples are in red and blue, respectively.

Table 7
$\mathcal{A}$-distance across domains on six tasks of the *Office-31* dataset.

	A→W	D→W	W→D	A→D	D→A	W→A	Average
$\mathcal{D}_s$ and $\mathcal{D}_\tau$	1.84	0.71	0.89	1.90	1.68	1.90	1.54
$\mathcal{D}_g$ and $\mathcal{D}_\tau$	0.45	0.33	0.51	0.57	1.20	1.31	0.73
$\mathcal{D}_{\hat{s}}$ and $\mathcal{D}_\tau$	0.64	0.51	0.52	0.65	1.31	1.39	0.84



(a) (b) (c)

Fig. 7. The images from left to right, separated by black solid lines, are source domain samples, target domain samples, and generated target domain samples, respectively. (a) Transfer task Ar→Cl from the *Office–Home* dataset. (b) Transfer task Synthetic→Real from the *VisDA* dataset. (c) Transfer task R→S from the *miniDomainNet* dataset.

Fig. 6 shows the t-SNE visualization (Van der Maaten & Hinton, 2008) of feature embeddings for samples from the six transfer tasks on *Office-31* (using a *ResNet-50* pretrained on *ImageNet*). As can be seen, embeddings of the generated target domain samples (in red) are close to those of the real target domain samples (in blue). Table 7 shows the $\mathcal{A}$-distances[5] (Ben-David, Blitzer, Crammer, & Pereira, 2006) (which measure distributional discrepancies) of $\mathcal{D}_s/\mathcal{D}_g/\mathcal{D}_{\hat{s}}$ to $\mathcal{D}_\tau$. As can be seen, compared with $\mathcal{D}_s$, $\mathcal{D}_g$ and $\mathcal{D}_{\hat{s}}$ are closer to $\mathcal{D}_\tau$ on all six tasks, implying that the samples generated by DCDM make it easier to transfer from the augmented source domain $\mathcal{D}_{\hat{s}}$ to the target domain $\mathcal{D}_\tau$ than the source domain $\mathcal{D}_s$.

Moreover, we use the generated target domain samples from the Ar→Pr task to perform the Ar→Cl task, and the generated target domain samples from the Ar→Cl task to perform the Ar→Pr task. The experimental results, showing a performance drop from 57.79% to 54.00% for the Ar→Cl task and from 77.65% to 68.69% for the Ar→Pr task, indicate that the augmented samples, which deviate from the target domain distribution, lead to negative transfer. Therefore, the transfer performance is influenced by the distance between the generated target samples and the real target domain samples, i.e., $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_g, \mathcal{D}_\tau)$, as shown in the right-hand side of Eq. (10).

Figs. 7(a), (b) and (c) show some generated images for the transfer tasks Ar→Cl, Synthetic→Real, and R→S on *Office–Home*, *VisDA-2017*, and *miniDomainNet*, respectively. According to the generated images, we can see that the styles of the generated samples are similar to those of the real target domain samples.

4.4. Ablation studies

In this section, we perform a number of ablation experiments on *Office-31*.

[5] The $\mathcal{A}$-distance is defined as $d_{\mathcal{A}}(\mathcal{D}_s, \mathcal{D}_t) = 2(1 - 2\nu)$, where ν is the error rate of a linear domain discriminator to distinguish samples from the two domains.

Table 8
Effect of class condition (denoted "Class") and domain guidance (denoted "Domain") on accuracy (%) of the *Office-31* dataset. The best is in bold.

Class	Domain	A→W	D→W	W→D	A→D	D→A	W→A	Average
✗	✗	93.84	98.78	**100.00**	95.78	77.72	75.13	90.21
✓	✗	94.48	**99.00**	**100.00**	96.92	79.09	76.88	91.06
✓	✓	**96.90**	98.91	**100.00**	**97.46**	**79.79**	**77.74**	**91.80**

(a) ELS+DCDM (w/o domain guidance). (b) ELS+DCDM.

Fig. 8. Generated target domain images for the transfer task A→W of the *Office-31* dataset.

Table 9
Transfer accuracies (%) for different combinations for controlling classes and domains on the *Office-31* dataset. The best is in bold.

Class	Domain	A→W	D→W	W→D	A→D	D→A	W→A	Average
Condition	Condition	96.86	**98.99**	**100.00**	96.59	79.23	77.32	91.50
Condition	Guidance	**96.90**	98.91	**100.00**	**97.46**	**79.79**	**77.74**	**91.80**
Guidance	Condition	93.96	98.36	**100.00**	93.17	72.88	73.34	88.62
Guidance	Guidance	90.82	96.73	99.00	90.56	70.86	68.65	86.10

4.4.1. Effects of domain guidance in DCDM

In this experiment, we study the effect of domain guidance. We consider the three combinations: (i) without class condition and domain guidance (i.e., ELS); (ii) with class condition but without domain guidance (denoted "ELS+DCDM (w/o domain guidance)"); (iii) with both class condition and domain guidance (i.e., ELS+DCDM).

Table 8 shows the results of six transfer tasks on *Office-31*. As can be seen, ELS+DCDM achieves better performance than ELS+DCDM (w/o domain guidance) on average, demonstrating the effectiveness of using domain guidance for generating samples. Additionally, Fig. 8 shows the target domain backpack images generated by ELS+DCDM (w/o domain guidance) and ELS+DCDM for the transfer task A→W. As can be seen, when training with only the target domain (w/o domain guidance), only black and gray backpacks can be generated (Fig. 8(a)). The reason is that the target domain only contains backpacks in those colors. In contrast, with the introduction of source domain samples and domain guidance, backpacks of various colors similar to the target domain distribution are generated (Fig. 8(b)). Therefore, the proposed domain guidance strategy could leverage information from the source domain to enhance the diversity of generated samples.

4.4.2. Effects of condition and guidance in DCDM

In this experiment, we study the different combinations of guidance and conditional methods for controlling classes and domains. We consider four combinations: (i) Both the classes and domains are controlled by conditions, with the label and domain embeddings added together and then fed to the conditional diffusion model in Algorithm 1; (ii) The classes are controlled by condition while the domains are controlled by guidance (i.e., the proposed DCDM); (iii) The classes are controlled by guidance while the domains are controlled by condition, with the classifier trained on the source domain data and target domain samples with pseudo-labels, and the domain labels are used as the condition $\mathbf{c}$ in Eq. (5); (iv) Both the classes and domains are controlled by guidance (i.e., the sum of gradients of the class classifier and domain classifier, which are trained separately).

Table 9 shows the testing accuracies on the six transfer tasks from *Office-31*. As can be seen, the combination that uses condition on classes and guidance on domains achieves the best performance, while controlling both classes and domains by condition is slightly inferior. Furthermore, when guidance is used to control the class, the performance is usually worse. One possible reason is that as some classes have limited samples,[6] the trained class classifier may not be good enough for guiding. On the other hand, domain classification is simply a binary classification problem, which is much easier. Thus, the trained domain classifier can still be good enough for guidance. Figs. 9(c) and 9(d) show generated images on the "backpack" class for the transfer task A→D when guidance is used to control the class. As can be seen, some of the images generated are not backpacks, leading to worse performance (Table 9).

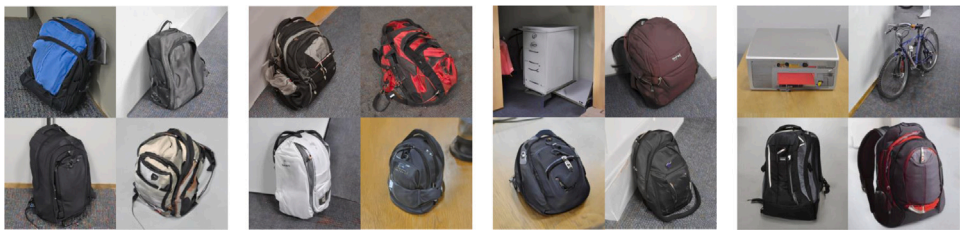
4.4.3. Effects of the number of generated samples

In this experiment, we show how the number of target domain samples generated affects the performance of ELS+DCDM on *Office-31*, *Office–Home*, *VisDA-2017*, and *miniDomainNet*. Fig. 10 shows the average accuracies w.r.t. the number of target domain samples generated per class. As can be seen, increasing the number of target domain samples generated boosts the performance. When no target domain sample is generated, ELS+DCDM degenerates to ELS, which performs worse than ELS+DCDM on all four datasets. Furthermore, when the number of target domain samples generated further increases, the performance saturates and may even slightly decrease. Hence, in the experiments above, the number of target domain samples generated for each class is set to 200, 200, 2000, and 200 for *Office-31*, *Office–Home*, *VisDA*, and *miniDomainNet*, respectively.

5. Conclusion

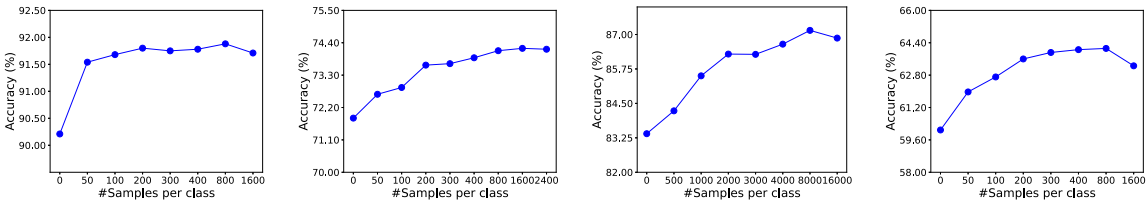
In this paper, we propose the DCDM method to generate high-fidelity samples for the target domain in UDA. The classes of generated samples are controlled by conditional diffusion models, while

[6] For example, domain DSLR (resp. Webcam) has only 16 (resp. 26) samples per class on average.



(a) (b) (c) (d)

Fig. 9. Generated target domain images for the transfer task A→D from the *Office-31* dataset. (a) The class and domain are both controlled by condition. (b) The class is controlled by condition while the domain is controlled by guidance. (c) The class is controlled by guidance while the domain is controlled by condition. (d) Both class and domain are controlled by guidance.



(a) *Office-31*. (b) *Office-Home*. (c) *VisDA-2017*. (d) *miniDomainNet*.

Fig. 10. Accuracy w.r.t. target domain samples generated per class on the *Office-31*, *Office–Home*, *VisDA-2017*, and *miniDomainNet* datasets.

the domain is guided by the domain classifier. DCDM can be integrated into any UDA model. Extensive experimental results demonstrate that DCDM achieves state-of-the-art performance. Compared with discrepancy-based and adversarial-based UDA methods, the proposed method requires an extra process for training and sampling the diffusion model. Also, the generated samples consume extra storage space. In our future work, we will study applying DCDM to other transfer learning settings such as multi-source domain adaptation (Wen, Chen, Xie, Liu, & Zheng, 2024).

CRediT authorship contribution statement

Yulong Zhang: Writing – review & editing, Writing – original draft, Methodology, Data curation. **Shuhao Chen:** Visualization, Software, Investigation. **Weisen Jiang:** Writing – review & editing, Supervision. **Yu Zhang:** Writing – review & editing, Validation, Funding acquisition, Formal analysis. **Jiangang Lu:** Writing – review & editing, Conceptualization. **James T. Kwok:** Writing – review & editing, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by NSFC key grant with grant no. 62136005 and NSFC general grant with grant no. 62076118.

Appendix. Generalization error bound

In the UDA scenario, the target domain is unlabeled. Suppose there are some labeled target domain samples D_g, and they are augmented into the source domain. We analyze the generalization error bound of UDA.

Suppose there are some labeled target domain samples from $\mathcal{T}$ merge into the source domain as the augmented source domain:

$$\hat{S} = (1-\alpha)S + \alpha\mathcal{T},$$

where $0 < \alpha < 1$ indicates the fraction of target domain samples added to the source domain.

According to Theorem 2 in Ben-David et al. (2010), the generalization error bound on the target domain with the augmented source domain can be defined as:

$$\mathcal{E}_\tau(h) \le \mathcal{E}_{\hat{s}}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\hat{S}, \mathcal{T}),$$

where $d_{\mathcal{H}\Delta\mathcal{H}}(\hat{S}, \mathcal{T})$ denotes the $\mathcal{H}\Delta\mathcal{H}$-distance between $\hat{S}$ and $\mathcal{T}$, i.e.,

$$d_{\mathcal{H}\Delta\mathcal{H}}(\hat{S}, \mathcal{T}) = 2 \sup_{h,h' \in \mathcal{H}} \left| \Pr_{x\sim\hat{S}}[h(x) \neq h'(x)] - \Pr_{x\sim\mathcal{T}}[h(x) \neq h'(x)] \right|.$$

Using the definition of the augmented source distribution, we have:

$$\Pr_{x\sim\hat{S}}[h(x) \neq h'(x)] = (1-\alpha) \Pr_{x\sim\hat{S}}[h(x) \neq h'(x)] + \alpha \Pr_{x\sim\mathcal{T}}[h(x) \neq h'(x)].$$

Thus, the divergence becomes:

$$\begin{aligned}
&\left| \Pr_{x\sim\hat{S}}[h(x) \neq h'(x)] - \Pr_{x\sim\mathcal{T}}[h(x) \neq h'(x)] \right| \\
=&\left| (1-\alpha) \Pr_{x\sim S}[h(x) \neq h'(x)] + \alpha \Pr_{x\sim\mathcal{T}}[h(x) \neq h'(x)] - \Pr_{x\sim\mathcal{T}}[h(x) \neq h'(x)] \right| \\
=&\left| (1-\alpha) \left(\Pr_{x\sim S}[h(x) \neq h'(x)] - \Pr_{x\sim\mathcal{T}}[h(x) \neq h'(x)] \right) \right| \\
=&(1-\alpha) \left| \Pr_{x\sim S}[h(x) \neq h'(x)] - \Pr_{x\sim\mathcal{T}}[h(x) \neq h'(x)] \right|.
\end{aligned}$$

Therefore, we have:

$$d_{\mathcal{H}\Delta\mathcal{H}}(\hat{S}, \mathcal{T}) = (1-\alpha) d_{\mathcal{H}\Delta\mathcal{H}}(S, \mathcal{T})$$

Since $0 < \alpha < 1$, we have:

$$d_{\mathcal{H}\Delta\mathcal{H}}(\hat{S}, \mathcal{T}) \le d_{\mathcal{H}\Delta\mathcal{H}}(S, \mathcal{T}).$$

and the equality holds only when $d_{\mathcal{H}\Delta\mathcal{H}}(\hat{S}, \mathcal{T}) = d_{\mathcal{H}\Delta\mathcal{H}}(S, \mathcal{T}) = 0$.

For overparameterized networks which are powerful enough to fitting training samples (Cybenko, 1989; Hornik, Stinchcombe, & White, 1989; Neyshabur, Tomioka, & Srebro, 2015; Zhang, Bengio, Hardt, Recht, & Vinyals, 2021), the losses $\mathcal{E}_s(h)$ and $\mathcal{E}_{\hat{s}}(h)$ will become negligible. Thus, the generalization bound with target domain augmentation may be smaller than that without target domain augmentation since $d_{\mathcal{H}\Delta\mathcal{H}}(\hat{S}, \mathcal{T}) \le d_{\mathcal{H}\Delta\mathcal{H}}(S, \mathcal{T})$. However, directly obtaining labeled target

samples is infeasible in the UDA scenario, making it reasonable to use generative models to generate labeled samples for the target domain. To this end, we propose the DCDM method to generate target domain samples using diffusion models.

Data availability

Data will be made available on request.

References

Abu-Mostafa, Y. S., Magdon-Ismail, M., & Lin, H.-T. (2012). *Learning from data*: *vol. 4*, AMLBook New York.

Bao, Y., Chang, S., & Barzilay, R. (2022). Learning stable classifiers by transferring unstable features. In *Int. conf. mach. learn.* (pp. 1483–1507). PMLR.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. W. (2010). A theory of learning from different domains. *Machine Learning*, *79*(1), 151–175.

Ben-David, S., Blitzer, J., Crammer, K., & Pereira, F. (2006). Analysis of representations for domain adaptation. In *Proc. adv. neural inf. process. syst., vol. 19* (pp. 137–144).

Cao, H., Tan, C., Gao, Z., Xu, Y., Chen, G., Heng, P.-A., et al. (2024). A survey on generative diffusion models. *IEEE Transactions on Knowledge and Data Engineering*.

Chen, W., Wang, H., Yang, S., Zhang, L., Wei, W., Zhang, Y., et al. (2023). Adapt anything: Tailor any image classifiers across domains and categories using text-to-image diffusion models. arXiv preprint arXiv:2310.16573.

Chen, J., Wu, X., Duan, L., & Gao, S. (2022). Domain adversarial reinforcement learning for partial domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(2), 539–553.

Croitoru, F.-A., Hondru, V., Ionescu, R. T., & Shah, M. (2023). Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–20.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, *2*(4), 303–314.

Dhaini, M., Berar, M., Honeine, P., & Van Exem, A. (2023). Unsupervised domain adaptation for regression using dictionary learning. *Knowledge-Based Systems*, *267*, Article 110439.

Dhariwal, P., & Nichol, A. (2021). Diffusion models beat GANs on image synthesis. In *Proc. adv. neural inf. process. syst. , vol. 34* (pp. 8780–8794).

Dosovitskiy, A. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

Fang, Y., Yap, P.-T., Lin, W., Zhu, H., & Liu, M. (2024). Source-free unsupervised domain adaptation: A survey. *Neural Networks*, Article 106230.

Foret, P., Kleiner, A., Mobahi, H., & Neyshabur, B. (2021). Sharpness-aware minimization for efficiently improving generalization. In *Proc. int. conf. learn. represent.*.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., et al. (2016). Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, *17*(1), 2096–2030.

Gao, Y., Chu, Z., Wang, H., & Stankovic, J. (2022). MiddleGAN: Generate domain agnostic samples for unsupervised domain adaptation. arXiv:2211.03144.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *Proc. int. conf. learn. represent.*.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., & Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, *13*(1), 723–773.

Gu, X., Sun, J., & Xu, Z. (2022). Unsupervised and semi-supervised robust spherical space domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–17.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proc. IEEE/CVF conf. comput. vis. pattern recognit.* (pp. 770–778).

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. In *Proc. adv. neural inf. process. syst., vol. 33* (pp. 6840–6851).

Hoffman, J., Tzeng, E., Park, T., Zhu, J.-Y., Isola, P., Saenko, K., et al. (2018). CyCADA: Cycle-consistent adversarial domain adaptation. In *Proc. int. conf. mach. learn. vol. 5* (pp. 1989–1998). Pmlr.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366.

Huang, F., Song, S., & Zhang, L. (2024). Gradient harmonization in unsupervised domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Jin, Y., Wang, X., Long, M., & Wang, J. (2020). Minimum class confusion for versatile domain adaptation. In *Proc. Eur. conf. comput. vis.* (pp. 464–480). Springer.

Li, K., Lu, J., Zuo, H., & Zhang, G. (2022). Dynamic classifier alignment for unsupervised multi-source domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, *35*(5), 4727–4740.

Li, H., Zheng, C., Xiao, Y., Wu, P., Geng, Z., Chen, X., et al. (2024). Debiased collaborative filtering with kernel-based causal balancing. In *Proc. int. conf. learn. represent.*.

Litrico, M., Del Bue, A., & Morerio, P. (2023). Guiding pseudo-labels with uncertainty estimation for source-free unsupervised domain adaptation. In *Proc. IEEE/CVF conf. comput. vis. pattern recognit.* (pp. 7640–7650).

Liu, W., Chen, C., Liao, X., Hu, M., Tan, Y., Wang, F., et al. (2024). Learning accurate and bidirectional transformation via dynamic embedding transportation for cross-domain recommendation. In *Proc. AAAI conf. artif. intell., vol. 38, no. 8* (pp. 8815–8823).

Liu, W., Zheng, X., Su, J., Zheng, L., Chen, C., & Hu, M. (2023). Contrastive proxy kernel stein path alignment for cross-domain cold-start recommendation. *IEEE Transactions on Knowledge and Data Engineering*.

Long, M., Cao, Y., Wang, J., & Jordan, M. (2015). Learning transferable features with deep adaptation networks. In *Proc. int. conf. mach. learn., vol. 1* (pp. 97–105). PMLR.

Long, M., Cao, Z., Wang, J., & Jordan, M. I. (2018). Conditional adversarial domain adaptation. In *Proc. adv. neural inf. process. syst., vol. 31* (pp. 1640–1650).

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., & Zhu, J. (2022). DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Proc. adv. neural inf. process. syst., vol. 35*.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., & Zhu, J. (2023). DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. arXiv:2211.01095.

Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

Na, J., Jung, H., Chang, H. J., & Hwang, W. (2021). FixBi: Bridging domain spaces for unsupervised domain adaptation. In *Proc. IEEE/CVF conf. comput. vis. pattern recognit.* (pp. 1094–1103).

Neyshabur, B., Tomioka, R., & Srebro, N. (2015). Norm-based capacity control in neural networks. In *Proc. conf. learn. theory* (pp. 1376–1401). PMLR.

Nichol, A. Q., & Dhariwal, P. (2021). Improved denoising diffusion probabilistic models. In *Proc. int. conf. mach. learn., vol. 139* (pp. 8162–8171). PMLR.

Odena, A., Olah, C., & Shlens, J. (2017). Conditional image synthesis with auxiliary classifier gans. In *Proc. int. conf. mach. learn.* (pp. 2642–2651). PMLR.

Oza, P., Sindagi, V. A., Sharmini, V. V., & Patel, V. M. (2023). Unsupervised domain adaptation of object detectors: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–24.

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., & Wang, B. (2019). Moment matching for multi-source domain adaptation. In *Proc. IEEE/CVF int. conf. comput. vis.* (pp. 1406–1415).

Peng, X., Usman, B., Kaushik, N., Hoffman, J., Wang, D., & Saenko, K. (2017). VisDA: The visual domain adaptation challenge. arXiv:1710.06924.

Rangwani, H., Aithal, S. K., Mishra, M., Jain, A., & Radhakrishnan, V. B. (2022). A closer look at smoothness in domain adversarial training. In *Proc. int. conf. mach. learn., vol. 162* (pp. 18378–18399). PMLR.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10684–10695).

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *Proc. int. conf. med. image comput. comput.-assist. intervent.* (pp. 234–241). Springer.

Saenko, K., Kulis, B., Fritz, M., & Darrell, T. (2010). Adapting visual category models to new domains. In *Proc. Eur. conf. comput. vis., vol. 6314* (pp. 213–226). Springer.

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*(1), 1–48.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., & Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. int. conf. mach. learn., vol. 3* (pp. 2256–2265). PMLR.

Sohn, K., Lee, H., & Yan, X. (2015). Learning structured output representation using deep conditional generative models. In *Proc. adv. neural inf. process. syst, vol. 28*.

Song, Y., Durkan, C., Murray, I., & Ermon, S. (2021). Maximum likelihood training of score-based diffusion models. In *Proc. adv. neural inf. process. syst, vol. 34* (pp. 1415–1428).

Song, J., Meng, C., & Ermon, S. (2021). Denoising diffusion implicit models. In *Proc. int. conf. learn. represent.*.

Sun, T., Lu, C., Zhang, T., & Ling, H. (2022). Safe self-refinement for transformer-based domain adaptation. In *Proc. IEEE/CVF conf. comput. vis. pattern recognit.* (pp. 7191–7200).

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(11), 2579–2625.

Vapnik, V. (1999). *The nature of statistical learning theory*. Springer science & business media.

Venkateswara, H., Eusebio, J., Chakraborty, S., & Panchanathan, S. (2017). Deep hashing network for unsupervised domain adaptation. In *Proc. IEEE/CVF conf. comput. vis. pattern recognit.* (pp. 5018–5027).

Villaizán-Vallelado, M., Salvatori, M., Carro, B., & Sanchez-Esguevillas, A. J. (2024). Graph neural network contextual embedding for deep learning on tabular data. *Neural Networks*, *173*, Article 106180.

Wang, H. (2024). Improving neural network generalization on data-limited regression with doubly-robust boosting. In *Proc. AAAI conf. artif. intell., vol. 38, no. 18* (pp. 20821–20829).

Wang, H., Fan, J., Chen, Z., Li, H., Liu, W., Liu, T., et al. (2023). Optimal transport for treatment effect estimation. In *Proc. adv. neural inf. process. syst., vol. 36*.

Wang, J., Wu, S., Yuan, Z., Tong, Q., & Xu, K. (2024). Frequency compensated diffusion model for real-scene dehazing. *Neural Networks*, *175*, Article 106281.

Wen, L., Chen, S., Xie, M., Liu, C., & Zheng, L. (2024). Training multi-source domain adaptation network by mutual information estimation and minimization. *Neural Networks*, *171*, 353–361.

Xia, H., Jing, T., & Ding, Z. (2023). Maximum structural generation discrepancy for unsupervised domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(3), 3434–3445.

Xie, M., Li, S., Zhang, R., & Liu, C. H. (2023). Dirichlet-based uncertainty calibration for active domain adaptation. In *Proc. int. conf. learn. represent.*.

Xu, R., Li, G., Yang, J., & Lin, L. (2019). Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proc. IEEE/CVF int. conf. comput. vis.*.

Yang, J., Liu, J., Xu, N., & Huang, J. (2023). Tvt: Transferable vision transformer for unsupervised domain adaptation. In *Proc. IEEE winter conf. appl. comput. vis.* (pp. 520–530).

Yang, G., Xia, H., Ding, M., & Ding, Z. (2020). Bi-directional generation for unsupervised domain adaptation. In *Proc. AAAI conf. artif. intell., vol. 34, no. 4* (pp. 6615–6622).

Yang, Q., Zhang, Y., Dai, W., & Pan, S. J. (2020). *Transfer learning*. Cambridge, U.K.: Cambridge Univ. Press.

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., et al. (2023). Diffusion models: A comprehensive survey of methods and applications. arXiv:2209.00796.

Ye, S., & Liu, F. (2024). Score mismatching for generative modeling. *Neural Networks*, *175*, Article 106311.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, *64*(3), 107–115.

Zhang, Y., Liu, T., Long, M., & Jordan, M. (2019). Bridging theory and algorithm for domain adaptation. In *Proc. int. conf. mach. learn.* (pp. 12805–12823).

Zhang, Y., Wang, Y., Jiang, Z., Liao, F., Zheng, L., Tan, D., et al. (2022). Diversifying tire-defect image generation based on generative adversarial network. *IEEE Transactions on Instrumentation and Measurement*, *71*, 1–12.

Zhang, Y., Wang, X., Liang, J., Zhang, Z., Wang, L., Jin, R., et al. (2023). Free lunch for domain adversarial training: Environment label smoothing. In *Proc. int. conf. learn. represent.*.

Zhang, Y., & Yang, Q. (2022). A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, *34*(12), 5586–5609.

Zhang, Y., Yao, Y., Chen, S., Jin, P., Jin, J., & Jiangang, L. (2024). Rethinking guidance information to utilize unlabeled samples: A label-encoding perspective. In *Proc. int. conf. mach. learn.*.

Zhao, S., Yue, X., Zhang, S., Li, B., Zhao, H., Wu, B., et al. (2022). A review of single-source deep unsupervised visual domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(2), 473–493.

Zhou, K., Yang, Y., Qiao, Y., & Xiang, T. (2021). Domain adaptive ensemble learning. *IEEE Transactions on Image Processing*, *30*, 8008–8018.

Zhu, Y., Zhuang, F., Wang, J., Ke, G., Chen, J., Bian, J., et al. (2021). Deep subdomain adaptation network for image classification. *IEEE Transactions on Neural Networks and Learning Systems*, *32*(4), 1713–1722.

Zhuang, Z., Zhang, Y., & Wei, Y. (2024). Gradual domain adaptation via gradient flow. In *Proc. int. conf. learn. represent.*.