Transfer Learning for Diffusion Models

Yidong Ouyang[1], **Liyan Xie**[2]*, **Hongyuan Zha**[3], **Guang Cheng**[1]

[1]Department of Statistics and Data Science, University of California, Los Angeles
[2]Department of Industrial and Systems Engineering, University of Minnesota Twin Cities
[3]School of Data Science, Chinese University of Hong Kong, Shenzhen
yidongouyang@g.ucla.edu; liyanxie@umn.edu; zhahy@cuhk.edu.cn; guangcheng@ucla.edu

Abstract

Diffusion models, a specific type of generative model, have achieved unprecedented performance in recent years and consistently produce high-quality synthetic samples. A critical prerequisite for their notable success lies in the presence of a substantial number of training samples, which can be impractical in real-world applications due to high collection costs or associated risks. Consequently, various finetuning and regularization approaches have been proposed to transfer knowledge from existing pre-trained models to specific target domains with limited data. This paper introduces the Transfer Guided Diffusion Process (TGDP), a novel approach distinct from conventional finetuning and regularization methods. We prove that the optimal diffusion model for the target domain integrates pre-trained diffusion models on the source domain with additional guidance from a domain classifier. We further extend TGDP to a conditional version for modeling the joint distribution of data and its corresponding labels, together with two additional regularization terms to enhance the model performance. We validate the effectiveness of TGDP on both simulated and real-world datasets.

1 Introduction

Diffusion models have achieved remarkable success in modeling data distributions and generating various types of synthetic data, such as images [13, 36, 17], videos [14], vision language [32, 33, 30], and time series [39]. However, their success heavily relies on the availability of a large number of training samples. In real-world applications, acquiring ample samples for specific tasks can be challenging due to the high costs associated with data collection or labeling, or the potential risks involved. Therefore, an important research question is how to effectively transfer knowledge from a pre-trained generative model in the source domain (using existing large-scale datasets) to a target domain (for specific tasks) where data is limited.

Training a generative model directly or finetuning a pre-trained generative model on limited data from the target domain often results in significant performance degradation due to overfitting and memorization. To address these issues, numerous studies have proposed methods in generative domain adaptation, including the GAN-based models [46, 45, 49, 1, 51, 28, 48, 9, 15, 43, 22, 50], diffusion-based model [25, 52, 44], etc. Specifically, approaches using diffusion models can be divided into two categories: finetuning lightweight adapters [25, 44] and finetuning with regularization [52]. Approaches involving finetuning lightweight adapters focus on adjusting only a subset of parameters in a pre-trained model. The primary challenge here is identifying which parameters to finetune. This process is typically heuristic and requires preliminary experiments to identify the most efficient parameters for adjustment. Additionally, the specific parameters to be finetuned can vary across

*Correspondence to: Liyan Xie, liyanxie@umn.edu.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

different neural network architectures. On the other hand, the challenge in incorporating regularization during the finetuning process is the heuristic design of the regularization term, which can significantly alter the optimization landscape. We refer to Appendix A for a more detailed discussion of existing literature.

In this work, we introduce a new approach, termed Transfer Guided Diffusion Process (TGDP), to transfer knowledge in the source domain generative model to the target domain with limited samples. Unlike finetuning-based methods that primarily use the pre-trained model as an initialization point, TGDP leverages the pre-trained model as a plug-and-play prior. We show that the score function for the diffusion model on the target domain is the score function on the source domain (which can be pre-trained) with additional guidance as shown by Theorem 3.1 and Theorem 3.3. The guidance network is related to the density ratio of the target and source domain data distributions. Consequently, we convert the original optimization problem for a diffusion model on the target domain into estimating the density ratio.

We utilize a domain classifier (binary classifier) along with samples from both domains to efficiently estimate the density ratio. Furthermore, we introduce two additional regularization terms for better training and calibration of the guidance network. These regularization terms are equivalent forms that the optimal guidance network should satisfy, ensuring they do not alter the original optimization problem. We validate the effectiveness of our approach through experiments on Gaussian mixture simulations and real electrocardiogram (ECG) data. Under both fidelity and utility evaluation criteria, TGDP consistently outperforms finetuning-based methods.

Our contributions can be summarized as follows.

- We introduce a new framework, the Transfer Guided Diffusion Process (TGDP), for transferring a pre-trained diffusion model from the source domain to the target domain.
- We extend TGDP to a conditional version for modeling the joint distribution of data and its corresponding labels, along with two additional regularization terms, which are important for practical applications and downstream tasks.
- TGDP demonstrates superior performance over finetuning-based methods on Gaussian mixture simulations and on benchmark electrocardiogram (ECG) data.

The rest of the paper is organized as follows. Section 2 reviews the setup of generative domain adaptation and the diffusion model. Section 3 introduces the proposed method and theoretically characterizes its effectiveness. Numerical results are given in Section 4. We conclude the paper in Section 5. All proofs and additional numerical experiments are deferred to the Appendix.

2 Problem Formulation and Preliminaries

2.1 Transfer Learning Problem Setup

Let $\mathcal{X}$ denote the data space and $\mathcal{Y}$ the label space. A domain corresponds to a joint distribution over $\mathcal{X}$ and $\mathcal{Y}$, denoted as p_{XY} for the *source* domain and q_{XY} for the *target* domain. The marginal distribution of data in the source and target domains are p_X and q_X, respectively. Suppose we have access to m (labeled) samples from the source domain $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m \sim p_{XY}$ and n (labeled) samples from the target domain $\mathcal{T} = \{(\mathbf{x}'_i, y'_i)\}_{i=1}^n \sim q_{XY}$. Typically, the source domain contains significantly more samples than the target domain, i.e., $n \ll m$. This setup reflects the common scenario where there is limited data available for specific tasks in the target domain, while abundant data is readily accessible and stored in the source domain.

The problem of interest is as follows. Given a pre-trained generative model p_θ for the data distribution p_X in the source domain, and a relatively small number of samples from the target domain, generative domain adaptation approaches aim to obtain a generative model that can generate synthetic samples following the target data distribution q_X. We will focus on diffusion generative models, given their great success in synthetic data generation. We first present the key idea of a carefully designed guidance network for the generation of $\mathbf{x}$ values only. Then, we extend the method to facilitate conditional generations so that we can generate paired samples with labels, $(\mathbf{x}, y)$, and can incorporate downstream classification tasks on the target domain.

2.2 Preliminaries of Diffusion Model

Diffusion models are characterized by their forward and backward processes. For illustrative purposes, we discuss the diffusion model trained on the source domain. The forward process involves perturbing the data distribution $p_X(\mathbf{x})$ by injecting Gaussian noise, as described by the following continuous-time equation [36]:

$$\mathrm{d}\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w},\ t \in [0, T], \tag{1}$$

where $\mathbf{w}$ is the standard Brownian motion, $\mathbf{f}(\cdot, t) : \mathbb{R}^d \to \mathbb{R}^d$ is a drift coefficient, and $g(\cdot) : \mathbb{R} \to \mathbb{R}$ is a diffusion coefficient. The marginal distribution of $\mathbf{x}_t$ at time t is denoted as $p_t(\mathbf{x}_t)$, and p_0 is the distribution of the initial value $\mathbf{x}_0$, which equals the true data distribution $p_X(\mathbf{x})$. For notational simplicity and provided it does not cause further confusion, we will refer to this diffusion process as p in the following, and we define $p(\mathbf{x}_t|\mathbf{x}_s), \forall s, t$, as the conditional distribution of $\mathbf{x}_t$ given the value $\mathbf{x}_s$. Similarly, for initial value $\mathbf{x}$ following the target domain distribution, we denote the corresponding probability measure induced by the above diffusion process (1) as q.

Then, we can reverse the forward process (1) for generation, defined as:

$$\mathrm{d}\mathbf{x}_t = \left[\mathbf{f}(\mathbf{x}_t, t) - g(t)^2\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t + g(t)\mathrm{d}\overline{\mathbf{w}}, \tag{2}$$

where $\overline{\mathbf{w}}$ is a standard Brownian motion when time flows backwards from T to 0, and $\mathrm{d}t$ is an infinitesimal negative time step. The key of the backward process is to estimate the score function of each marginal distribution, $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$, then the generation can be performed by discretizations of (2) [13, 36]. Score Matching [16, 40, 35] are proposed to train a neural network $\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$ (parameterized by $\boldsymbol{\phi}$) to estimate the score:

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{\lambda(t)\mathbb{E}_{p_t(\mathbf{x}_t)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\|_2^2\right]\right\}, \tag{3}$$

where $\lambda(t) : [0, T] \to \mathbb{R}_{>0}$ is a positive weighting function, t is uniformly sampled over $[0, T]$. One commonly adopted forward process is choosing an affine $\mathbf{f}(\mathbf{x}, t) = -\frac{1}{2}\beta(t)\mathbf{x}$ and $g(t) = \sqrt{\beta(t)}$, which yields the Gaussian transition distribution $p(\mathbf{x}_t|\mathbf{x}_s) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta(t)}\mathbf{x}_s, \beta(t)\mathbb{I})$, $t > s$, with $\beta(t) : [0, T] \to (0, 1)$ as a variance schedule. This is the Variance Preserving Stochastic Differential Equation (VP SDE) that we use in the numerical Section 4.

Several works on image generation [4, 5] and inverse problem [7] extends Score Matching to Conditional Score Matching, i.e.,

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{\lambda(t)\mathbb{E}_{p_t(\mathbf{x}_t, y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|y)\|_2^2\right]\right\}, \tag{4}$$

where $p_t(\mathbf{x}_t|y)$ is the conditional distribution of perturbed data $\mathbf{x}_t$ given corresponding label y.

3 Transfer Guided Diffusion Process

In this section, we introduce the proposed Transfer Guided Diffusion Process (TGDP) that leverages a pre-trained diffusion model – trained on the source domain data – to generate data in the target domain. The proposed approach is orthogonal to and different from the existing fine-tuning type methods. We introduce the additional guidance in Section 3.1. The methods for calculating the guidance are provided in Section 3.2. We extend our framework to the conditional diffusion model in Section 3.3 and we propose two regularization terms for enhancing the performance of our method in Section 3.4. All proofs are deferred to Appendix C.

3.1 Methodology Formulation

This subsection outlines the process of transferring knowledge from a diffusion generative model pre-trained using the source domain data $\mathcal{S}$ for generating samples that match the underlying distribution of target domain sample $\mathcal{T}$. The simplest non-transfer type approach involves directly training a diffusion model on samples $\mathcal{T}$ from the target domain by denoising Score Matching as described by Eq (3) or Eq (4). However, since we assume only a limited amount of data is accessible on the target domain, directly learning from the target domain is unlikely to yield an effective generative model.

Several studies propose to finetune the pre-trained diffusion model to alleviate the challenges caused by limited data and make use of acquired knowledge [25, 42, 53]. These methods typically design different strategies, such as adapters, to avoid finetuning all weights in a pre-trained model. However, these approaches generally use the pre-trained diffusion model from the source domain only as initial weights. Our method offers a different way for better utilization of the acquired knowledge.

Our proposed method is inspired by the key observation detailed in the following Theorem 3.1. Intuitively, the score function $\nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)$ for the target domain differs from the score function $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ of the source domain by a term related to the density ratio function q_X/p_X. We refer to this differing term as a guidance term in the following Theorem.

Theorem 3.1. *Consider two diffusion models on the source and target domain, denoted as p and q, respectively. Let the forward process on the target domain be identical to that on the source domain, $q(\mathbf{x}_t|\mathbf{x}_0) = p(\mathbf{x}_t|\mathbf{x}_0)$, and $\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t, t)$ is the score estimator in the target domain:*

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q_t(\mathbf{x}_t)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t)\|_2^2 \right] \right\}, \tag{5}$$

then we have

$$\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t, t) = \underbrace{\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)}_{\substack{\textit{pre-trained model} \\ \textit{on source}}} + \underbrace{\nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)} \left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right]}_{\textit{guidance}}. \tag{6}$$

Based on Eq (6), instead of solving $\mathbf{s}_{\boldsymbol{\phi}^*}$ from the limited training samples on the target domain, we construct $\mathbf{s}_{\boldsymbol{\phi}^*}$ by combing the pre-training score estimator and the guidance based on a binary classifier of source and target domain samples (detailed in Section 3.2). We comment on some potential advantages of this simple yet effective idea. First of all, we do not need to fine-tune the pre-trained diffusion model on the source domain, with the corresponding computation shifted to training the guidance network which is essentially a classifier. Second, the guidance network can be effectively estimated by a domain classifier using data from both the source and target domains. There is also great flexibility in constructing this guidance network due to the extensive literature on classification problems and density ratio estimation approaches. Additionally, the sample complexity for training a generative model could be much larger than a discriminative model, since the generative model needs to recover the full spectrum of target data distribution, while a domain classifier only needs to distinguish whether the sample is from the source or target distribution.

3.2 Learning Guidance Network

We calculate the guidance for the diffusion model on the target domain as defined in the second term of Eq (6) via two steps. In the first step, we estimate the density ratio $q(\mathbf{x}_0)/p(\mathbf{x}_0)$ by training a classifier $c_{\boldsymbol{\omega}}(\mathbf{x}) : \mathcal{X} \to [0, 1]$ to distinguish samples from the source and target domains. We adopt the typical logistic loss as follows:

$$\boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\omega}} \left\{ -\frac{1}{m} \sum_{\mathbf{x}_i \sim p} \log c_{\boldsymbol{\omega}}(\mathbf{x}_i) - \frac{1}{n} \sum_{\mathbf{x}'_i \sim q} \log(1 - c_{\boldsymbol{\omega}}(\mathbf{x}'_i)) \right\}. \tag{7}$$

Then, the density ratio $q(\mathbf{x}_0)/p(\mathbf{x}_0)$ can be estimated as $(1 - c_{\boldsymbol{\omega}^*}(\mathbf{x}_0))/c_{\boldsymbol{\omega}^*}(\mathbf{x}_0)$, and it can be shown that the optimal solution to the population counterpart of Eq (7) is exactly the true likelihood ratio [38]. It is worthwhile mentioning that we may only use a subset of source domain samples to learn the classifier $c_{\boldsymbol{\omega}}$ to alleviate the unbalanced sample sizes, and we could also adopt modern density ratio estimators to improve the accuracy [31]. After learning the density ratio $q(\mathbf{x}_0)/p(\mathbf{x}_0)$, the second step is to calculate the expectation $\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}[q(\mathbf{x}_0)/p(\mathbf{x}_0)]$ using Monte Carlo simulation. Since it is hard to sample from $q(\mathbf{x}_0|\mathbf{x}_t)$, we use the following equivalent formulation to get the value instead. This trick has also been used in previous work such as the Appendix H in [23].

Lemma 3.2. *For a neural network $h_{\boldsymbol{\psi}}(\mathbf{x}_t, t)$ parameterized by $\boldsymbol{\psi}$, define the objective*

$$\mathcal{L}_{\textit{guidance}}(\boldsymbol{\psi}) := \mathbb{E}_{p(\mathbf{x}_0, \mathbf{x}_t)} \left[\left\| h_{\boldsymbol{\psi}}(\mathbf{x}_t, t) - \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right\|_2^2 \right], \tag{8}$$

then its minimizer $\boldsymbol{\psi}^* = \arg\min_{\boldsymbol{\psi}} \mathcal{L}_{\textit{guidance}}(\boldsymbol{\psi})$ satisfies:

$$h_{\boldsymbol{\psi}^*}(\mathbf{x}_t, t) = \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)} \left[q(\mathbf{x}_0)/p(\mathbf{x}_0) \right].$$

By Lemma 3.2, we estimate the value $\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[q(\mathbf{x}_0)/p(\mathbf{x}_0)\right]$ using the guidance network $h_{\boldsymbol{\psi}^*}$ solved by minimizing the objective function $\mathcal{L}_{\text{guidance}}(\boldsymbol{\psi})$, which can be approximated by easy sampling from the joint distribution $p(\mathbf{x}_0, \mathbf{x}_t)$. Combine the above steps together, the estimated score function for the diffusion generative model on target domain q_X can be calculated as follows:

$$\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t, t) = \underbrace{\nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t)}_{\substack{\text{pre-trained model} \\ \text{on source}}} + \underbrace{\nabla_{\mathbf{x}_t} \log h_{\boldsymbol{\psi}^*}(\mathbf{x}_t, t)}_{\text{guidance network}}. \tag{9}$$

3.3 Extension to the Conditional Version

The approach outlined above is for generating the sample $\mathbf{x}$ in the target domain. In this section, we extend the idea to the conditional generation task. Such extension is essential when the label sets in the source and target domain are different since, in such cases, we usually rely on the conditional diffusion model for sampling [18, 21]. We first present the following theorem, which is an analog to Theorem 3.1 within the context of conditional score matching.

Theorem 3.3. *Assume $\mathbf{x}_t$ and y are conditional independent given $\mathbf{x}_0$ in the forward process, i.e., $p(\mathbf{x}_t|\mathbf{x}_0, y) = p(\mathbf{x}_t|\mathbf{x}_0)$, $\forall t \in [0, T]$, and let the forward process on the target domain be identical to that on the source domain $q(\mathbf{x}_t|\mathbf{x}_0) = p(\mathbf{x}_t|\mathbf{x}_0)$, and $\boldsymbol{\phi}^*$ is the optimal solution for the conditional diffusion model trained on target domain $q(\mathbf{x}_0, y)$, i.e.,*

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q_t(\mathbf{x}_t, y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t|y)\|_2^2 \right] \right\}, \tag{10}$$

then

$$\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t, y, t) = \underbrace{\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|y)}_{\substack{\textit{pre-trained conditional model} \\ \textit{on source}}} + \underbrace{\nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t, y)} \left[\frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} \right]}_{\textit{conditional guidance}}. \tag{11}$$

The key difference is we need to estimate the joint density ratio between the source and target domain. We can extend the density ratio estimator in Section 3.2 for estimating joint density ratio, i.e., also feed the label y into the classifier $c_{\boldsymbol{\omega}}(\mathbf{x}, y)$. The corresponding Lemma and its proof for the conditional version of Lemma 3.2 can be found in Appendix C.3. We further provide a detailed discussion about how to extend this conditional guidance to text-to-image generation tasks and when the source and target domain contain different class labels in Appendix B.

3.4 Additional Regularizations in Practical Implementations

In this subsection, we provide two additional regularization terms in our final objective function, to enhance the performance of the proposed scheme.

Cycle Regularization In the approaches described above, after obtaining the classifier network $c_{\boldsymbol{\omega}^*}$, calculation of the additional guidance $\nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}[q(\mathbf{x}_0)/p(\mathbf{x}_0)]$(or $\nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t, y)}[q(\mathbf{x}_0, y)/p(\mathbf{x}_0, y)]$ for conditional generation) *only* utilizes the data from source domain. In this section, we provide an enhancement in which the limited data from the target domain can also be utilized to improve the training of the guidance network $h_{\boldsymbol{\psi}}$.

Notice that (with detailed derivation given in Appendix C.5)

$$\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)} \left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right] = \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)} \left[\frac{q_t(\mathbf{x}_t)}{p_t(\mathbf{x}_t)} \right], \tag{12}$$

where recall $p_t(\mathbf{x}_t)$ and $q_t(\mathbf{x}_t)$ are the marginal distribution at time t for source and target distributions, respectively. A similar idea to Theorem 3.2 implies that we can learn the guidance network by solving the following optimization problem as well:

$$\boldsymbol{\psi}^* = \arg\min_{\boldsymbol{\psi}} \mathcal{L}_{\text{cycle}} := \mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_t)} \left[\left\| h_{\boldsymbol{\psi}}(\mathbf{x}_t, t) - \frac{q_t(\mathbf{x}_t)}{p_t(\mathbf{x}_t)} \right\|_2^2 \right]. \tag{13}$$

Moreover, in order to estimate the density ratio for marginal distributions at time t between the target and source data distribution, we train a time-dependent classifier $c_{\boldsymbol{\omega}}(\mathbf{x}, t)$ to distinguish samples from

source domain $p_t(\mathbf{x})$ and target domain $q_t(\mathbf{x})$ by the logistic loss as follow:

$$\boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\omega}} \left\{ -\frac{1}{m} \sum_{\mathbf{x}_0 \sim p} \sum_{\mathbf{x}_t | \mathbf{x}_0} \log c_{\boldsymbol{\omega}}(\mathbf{x}_t, t) - \frac{1}{n} \sum_{\mathbf{x}_0 \sim q} \sum_{\mathbf{x}_t | \mathbf{x}_0} \log(1 - c_{\boldsymbol{\omega}}(\mathbf{x}_t, t)) \right\},$$

where m, n are the number of training samples in source and target domains. The density ratio $q_t(\mathbf{x}_t)/p_t(\mathbf{x}_t)$ can then be calculated by $(1 - c_{\boldsymbol{\omega}^*}(\mathbf{x}_t, t))/(c_{\boldsymbol{\omega}^*}(\mathbf{x}_t, t))$.

Consistency Regularization Motivated by the fact that an optimal guidance network should recover the score in the target domain, we further use score matching in the target domain as the Consistency Regularization $\mathcal{L}_{\text{consistence}}$ to learn the guidance network better.

$$\begin{aligned} \boldsymbol{\psi}^* &= \arg\min_{\boldsymbol{\psi}} \mathcal{L}_{\text{consistence}} \\ &:= \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q(\mathbf{x}_0)} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \left[\| \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0) + \nabla_{\mathbf{x}_t} \log h_{\boldsymbol{\psi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0) \|_2^2 \right] \right\}. \end{aligned} \tag{14}$$

Combining these two additional regularization terms together with the original guidance loss (19), the final learning objective for the guidance network can be described as follows:

$$\boldsymbol{\psi}^* = \arg\min_{\boldsymbol{\psi}} \{ \mathcal{L}_{\text{guidance}} + \eta_1 \, \mathcal{L}_{\text{cycle}} + \eta_2 \, \mathcal{L}_{\text{consistence}} \}, \tag{15}$$

where $\eta_1, \eta_2 \geq 0$ are hyperparameters that control the strength of additional regularization, which also enhances the flexibility of our solution scheme. We summarize the Algorithm of TGDP in Appendix D.1. We provide the ablation studies that demonstrate the effectiveness of these two regularizations in Appendix D.2 and we also empirically show only adopt $\mathcal{L}_{\text{consistence}}$ to optimize the guidance network is not good enough because of the limited data from the target distribution.

Remark 3.4 (Discussion about related guidance). Classifier guidance has become a common trick in recent research [36, 8, 3, 6]. We highlight that, under the transfer learning framework, the guidance proposed in our work is the optimal guidance since the resulting score function matches the oracle score on the target domain. On the contrary, vanilla versions of classifier guidance utilizing a domain classifier cannot generate samples that exactly follow target distribution. Indeed, for a pre-trained domain classifier $c_{\boldsymbol{\omega}}$, vanilla domain classifier guidance formulates the source for generation as follows:

$$\begin{aligned} \mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t, t) &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) - \nabla_{\mathbf{x}_t} \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)} \left[\log(1 - c_{\boldsymbol{\omega}}(\mathbf{x}_0)) \right] \\ &\neq \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)} \left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right] \quad \text{(correct form proven in Theorem 3.1)} \\ &= \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)} \left[\frac{1 - c_{\boldsymbol{\omega}}(\mathbf{x}_0))}{c_{\boldsymbol{\omega}}(\mathbf{x}_0)} \right]. \end{aligned}$$

4 Experiments

In this section, we present empirical evidence demonstrating the efficacy of the proposed Transfer Guided Diffusion Process (TGDP) on limited data from a target domain. In Section 4.1, we conduct proof-of-concept experiments using a Gaussian mixture model to showcase that the guidance network of TGDP can successfully steer the pre-trained diffusion model toward the target domain. In Section 4.2, we illustrate the effectiveness of TGDP using a real-world electrocardiogram (ECG) dataset.

4.1 Simulation Results

Experimental setup We begin with a Gaussian mixture model where $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \{-1, 1\}$. On both domains, the marginal distribution for label y is uniform in $\mathcal{Y}$, and the conditional distribution of features is $\mathbf{x}|y \sim \mathcal{N}(y\boldsymbol{\mu}, \sigma^2 \mathbb{I}_d)$, where $\boldsymbol{\mu} \in \mathbb{R}^d$ is non-zero, and $\mathbb{I}_d$ is the d dimensional identity covariance matrix. We let $\boldsymbol{\mu} = \boldsymbol{\mu}_{\text{S}}$ on the source domain and $\boldsymbol{\mu} = \boldsymbol{\mu}_{\text{T}}$ on the target domain, with $(\boldsymbol{\mu}_{\text{S}})^\top \boldsymbol{\mu}_{\text{T}} = 0$. Under such case, the marginal distribution of $\mathbf{x}$ on the source domain p_X is a Gaussian mixture, for convenience we denote it as $0.5\mathcal{N}(\boldsymbol{\mu}_{\text{S}}, \sigma^2\mathbb{I}) + 0.5\mathcal{N}(-\boldsymbol{\mu}_{\text{S}}, \sigma^2\mathbb{I})$, and the marginal feature of target distribution q_X is $0.5\mathcal{N}(\boldsymbol{\mu}_{\text{T}}, \sigma^2\mathbb{I}) + 0.5\mathcal{N}(-\boldsymbol{\mu}_{\text{T}}, \sigma^2\mathbb{I})$. We let $d = 2$, $\boldsymbol{\mu}_{\text{S}} = [0.5, 0.5]$, $\boldsymbol{\mu}_{\text{T}} = [0.5, -0.5]$, $\sigma^2 = 0.1$, and draw $m = 10000$ samples from source domain p_X, and $n = 10$, 100, 1000 samples from target domain q_X, respectively.

Implementation details and Baselines We adopt the default Variance Preserving (VP) SDE in [36] with a linear schedule, i.e., $q(\mathbf{x}_t|\mathbf{x}_0) = p(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t|\alpha_t\mathbf{x}_0, \sigma_t^2\boldsymbol{I}\right)$ with α_t and σ_t being:

$$\alpha_t = -\frac{\beta_1 - \beta_0}{4}t^2 - \frac{\beta_0}{2}t, \quad \sigma_t = \sqrt{1-\alpha_t^2},$$

with $\beta_0 = 0.1, \beta_1 = 20$. We adopt 5-layer MLP with hidden sizes of $[512, 512, 512, 512, 256]$ and SiLU activation function as the diffusion model. We train the diffusion model on data from the source domain for 100 epochs using the Adam optimizer with a learning rate of $1\mathrm{e}^{-4}$ and batch size of 4096. The guidance network is a 4-layer MLP with 512 hidden units and SiLU activation function. We train the guidance network 20 epochs for our TGDP and train a vanilla diffusion model or finetune the diffusion model target domain 50 epochs. For generation, we adopt DPM-Solver [24] with a second-order sampler and a diffusion step of 25. We compare TGDP with the following baseline methods: 1) Vanilla Diffusion: directly training from target domain; 2) Finetune Diffusion: finetuning all weights of a pre-trained diffusion model on target distribution [2].

Experimental results We first demonstrate the effectiveness of guidance in Figure 1 under the above setup. Figure 1 (a) plots the source samples, while Figure 1 (b) shows the target samples under different sample sizes $n = 10, 100, 1000$. Figure 1 (c-e) illustrates the generated target samples via different methods, respectively. It can be seen that the samples generated via the proposed TGDP approach share similar patterns with the target distribution and two mixture components are more obvious as compared with other baseline methods. Furthermore, since the true data distribution of the target domain is known, we calculate the average likelihood of samples generated by each method as demonstrated in Table 1 for quantitative evaluation and comparison.

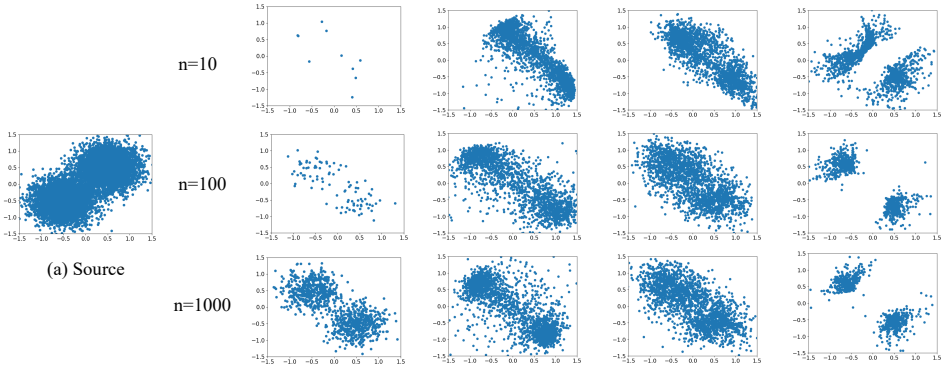


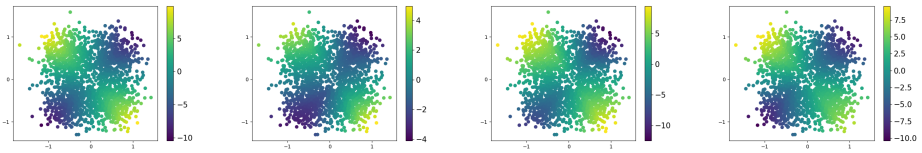
Figure 1: An illustration of the effectiveness of TGDP on simulations with 10/100/1000 target samples, respectively.

Table 1: Quantitative evaluation of TGDP on simulations. Training on 10K samples from the source domain and $n = 10, 100, 1000$ numbers on the target domain, respectively. TGDP achieves the highest average likelihood under target distribution.

	Average likelihood		
	n=10	n=100	n=1000
Vanilla Diffusion	0.145	0.253	0.328
Finetune	0.290	0.329	0.335
TGDP	**0.417**	**0.627**	**0.673**

[2]It is worthwhile mentioning that the reason we do not compared with the works that finetune partial weights in a pre-trained diffusion model [44] is their results are usually worse or comparable with method that finetunes all weights, the implementation of [25] are not available, and the regularization proposed by [52] is only valid for image data.

As a sanity check, we also look at the sensitivity of the learned density ratio estimator (through the classifier network (7)) regarding different sizes of target samples. As shown in Figure 2, even with only 10 samples from the target domain (and 10 samples from the source domain for class balance sampling), we can accurately estimate the landscape of density ratio (although the magnitude of the estimated ratio is not entirely accurate when the number of target samples equal 10).



(a) Oracle (b) 10 target (c) 100 target (d) 1000 target

Figure 2: An ablation study of the sensitivity of density ratio estimator.

4.2 ECG Data

In this section, we demonstrate the effectiveness of the proposed guidance on the benchmark of electrocardiogram (ECG) data. We first provide the standard synthetic quality and diversity evaluation in Section 4.2.1. Then, we utilize downstream classification tasks to further evaluate the effectiveness of TGDP in Section 4.2.2. We follow the setup of existing benchmarks on biomedical signal processing [37] that regard PTB-XL dataset [41] as the source domain and ICBEB2018 dataset [27] as the target domain. PTB-XL dataset contains 21,837 clinical 12-lead ECG recordings of 10 seconds length from 18,885 unique patients. A 12-lead ECG refers to the 12 different perspectives of the heart's electrical activity that are recorded. Moreover, the PTB-XL dataset is a multi-label dataset with 71 different statements (label). ICBEB2018 dataset [27] comprises 6877 12-lead ECGs lasting between 6 and 60 seconds. Each ECG record is categorized into one of nine classes, which is a subset of labels in the PTB-XL dataset. We randomly select 10% samples as limited target distribution by stratified sampling preserving the overall label distribution in each fold following [41]. We use the data from PTB-XL dataset and ICBEB2018 dataset at a sampling frequency of 100 Hz, which means 100 samples per second. We include more implementation details in Appendix D.3.

4.2.1 Synthetic Quality and Diversity Evaluation

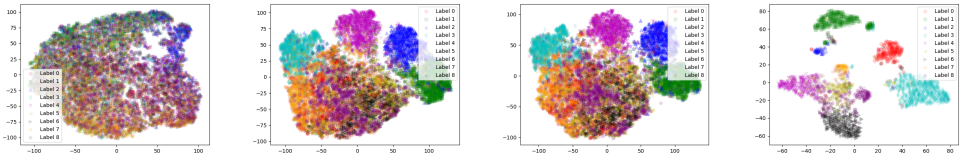
Baseline method We compare TGDP with the following baseline methods to demonstrate the effectiveness of TGDP. 1) Learn a generative model directly *(Vanilla Diffusion)*: The vanilla way is to learn a generative model directly on limited samples from the target domain. 2) Leveraging the pre-trained generative model from source domain *(Finetune Generator)*: Since the label set of the target domain is a subset of that in the source domain, a preliminary solution is to utilize the pre-trained diffusion model to generate samples with labels in the target domain.

Experimental results In Table 2, we compare the generation performance on the target domain using two metrics. The first criterion is the widely used Frechet Inception Distance (FID) [12] to evaluate the quality of synthetic data, which calculates the Wasserstein-2 distance between the real data and the synthetic data on the feature space. We use the pre-trained classifier on the target domain as the feature extractor, i.e., xresnet1d50 [37]. The second metric is the coverage [26] that evaluates the diversity of the synthetic data. It is defined as the ratio of real records that have at least one fake (synthetic) record in its sphere. The higher the coverage is, the more diverse the synthetic data are.

From Table 2, we see that TGDP achieves better performance than baseline methods on two criteria, which demonstrates the effectiveness of TGDP on generative transfer learning in scenarios with limited data. Moreover, TGDP has fewer parameters to be trained and less training time. We also demonstrate the T-SNE of the generated ECG data in Figure 3.

4.2.2 TGDP for Downstream Task

In Section 4.2.1, we illustrate that TGDP is capable of generating samples that adhere to the joint distribution of data and labels in the target domain and is diverse enough. In this subsection, we



(a) Vanilla Diffusion (b) Finetune Generator (c) TGDP (d) Real

Figure 3: T-SNE of the generated ECG data.

Table 2: The effectiveness of TGDP on ECG benchmark under synthetic quality and diversity criteria.

Method	Diversity (↑)	FID (↓)	Number of Parameters	Training Time
Vanilla Diffusion	0.37	11.01	50.2M	1h
Finetune Generator	0.47	12.26	50.2M	40min
TGDP	**0.53**	**10.46**	**2.8M**	30min

further investigate whether utilizing TGDP to acquire a generative model for the target domain yields superior performance compared to existing transfer learning pipelines.

Baseline method First of all, we can utilize the generative model learned in Section 4.2.1 to generate sufficient samples. Incorporated with the original limited sample from the target domain, we can train the classifier, which we still denoted as *Vanilla Diffusion*, *Finetune Generator*, and TGDP, respectively. Moreover, we have the following baseline methods. Directly train a classifier on target domain *(Vanilla Classifier)*: Utilizing the limited data from the target domain, a vanilla classifier can be obtained. Finetune pre-trained classifier *(Finetune Classifier)*: Instead of training a classifier from scratch on the target domain, the parameters of the classifier trained on the source domain are adjusted by using the limited data from the target domain. To verify the effectiveness of the generative model, we demonstrate that it improves the performance of the learned classifier in the following.

Experimental results We adopt the same evaluation criteria as ECG benchmark [37], i.e., Macro-averaged area under the receiver operating characteristic curve (AUC), Macro-averaged F_β-score ($\beta = 2$), where $F_\beta = \frac{(1+\beta^2)\cdot \text{TP}}{(1+\beta^2)\cdot \text{TP} + \beta^2 \cdot \text{FN} + \text{FP}}$, and Macro-averaged G_β-score with $\beta = 2$, where $G_\beta = \frac{\text{TP}}{\text{TP} + \text{FP} + \beta \cdot \text{FN}}$. In Table 3, TGDP outperforms baseline methods across three evaluation criteria, showcasing its effectiveness in transfer for diffusion model with limited data.

Table 3: The effectiveness of TGDP on ECG benchmark for downstream task. We provide 95% confidence intervals via empirical bootstrapping used by [37]. 0.906(03) stands for 0.906 ± 0.003.

Method	AUC	$F_{\beta=2}$	$G_{\beta=2}$
Vanilla Classifier	0.906(03)	0.674(06)	0.433(06)
Finetune Classifier	0.941(05)	0.747(08)	0.521(10)
Vanilla Diffusion	0.932(05)	0.718(09)	0.464(09)
Finetune Generator	0.941(04)	0.761(10)	0.528(12)
TGDP	**0.953**(05)	**0.773**(11)	**0.534**(11)

5 Conclusion

In this work, we propose a novel framework, Transfer Guided Diffusion Process (TGDP), for transferring a source-domain diffusion model to the target domain which consists of limited data. Instead of reducing the finetuning parameters or adding regularization for finetuning, TGDP proves the optimal diffusion model on the target domain is the pre-trained diffusion model on the source domain with additional guidance. TGDP outperforms existing methods on Gaussian mixture simulations and electrocardiogram (ECG) data benchmarks.

Limitations and broader impact Overall, this research presents a promising direction for leveraging pre-trained diffusion models to tackle new tasks. The proposed method, TGDP, has potential applications in a wide range of tasks where domain shift exists. A limitation of this study is the lack of empirical validation regarding TGDP's performance on language vision tasks, which we have earmarked for future exploration. Since we propose a generic algorithm for transferring knowledge to new tasks, this technique could enable people to train Deepfakes for disinformation better. Our approach hinges on the efficacy of detection methods in mitigating negative societal consequences.

6 Acknowledgement

Hongyuan Zha was supported in part by the Shenzhen Key Lab of Crowd Intelligence Empowered Low-Carbon Energy Network (No. ZDSYS20220606100601002). Guang Cheng was partially sponsored by NSF – SCALE MoDL (2134209), NSF – CNS (2247795), Office of Naval Research (ONR N00014-22-1-2680) and CISCO and Optum AI Research Grants.

References

[1] Aibek Alanov, Vadim Titov, and Dmitry P. Vetrov. Hyperdomainnet: Universal domain adaptation for generative adversarial networks. *ArXiv*, abs/2210.08884, 2022.

[2] Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based conditional ecg generation with structured state space models. *Computers in biology and medicine*, 163:107115, 2023.

[3] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 843–852, 2023.

[4] Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schonlieb, and Christian Etmann. Conditional image generation with score-based diffusion models. *ArXiv*, abs/2111.13606, 2021.

[5] Chen-Hao Chao, Wei-Fang Sun, Bo-Wun Cheng, Yi-Chen Lo, Chia-Che Chang, Yu-Lun Liu, Yu-Lin Chang, Chia-Ping Chen, and Chun-Yi Lee. Denoising likelihood score matching for conditional score-based data generation. In *ICLR*, 2022.

[6] Hyungjin Chung, Jeongsol Kim, Michael T. McCann, Marc Louis Klasky, and J. C. Ye. Diffusion posterior sampling for general noisy inverse problems. *ArXiv*, abs/2209.14687, 2022.

[7] Agnimitra Dasgupta, Javier Murgoitio-Esandi, Deep Ray, and Assad Oberai. Conditional score-based generative models for solving physics-based inverse problems. In *NIPS workshop, Deep Learning and Inverse Problems*, 2023.

[8] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *ArXiv*, abs/2105.05233, 2021.

[9] Yuxuan Duan, Li Niu, Y. Hong, and Liqing Zhang. Weditgan: Few-shot image generation via latent space relocation. *ArXiv*, abs/2305.06671, 2023.

[10] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, P. Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *ArXiv*, abs/2305.16381, 2023.

[11] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *ArXiv*, abs/2208.01618, 2022.

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Neural Information Processing Systems*, 2017.

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

[14] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. *ArXiv*, abs/2204.03458, 2022.

[15] Xingzhong Hou, B. Liu, Shuai Zhang, Lulin Shi, Zite Jiang, and Haihang You. Dynamic weighted semantic correspondence for few-shot image generative adaptation. *Proceedings of the 30th ACM International Conference on Multimedia*, 2022.

[16] Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *J. Mach. Learn. Res.*, 6:695–709, 2005.

[17] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *ArXiv*, abs/2206.00364, 2022.

[18] Levon Khachatryan, Andranik Movsisyan, Vahram Tadevosyan, Roberto Henschel, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Text2video-zero: Text-to-image diffusion models are zero-shot video generators. *ArXiv*, abs/2303.13439, 2023.

[19] Nupur Kumari, Bin Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1931–1941, 2022.

[20] Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, P. Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback. *ArXiv*, abs/2302.12192, 2023.

[21] Alexander C. Li, Mihir Prabhudesai, Shivam Duggal, Ellis L Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. *ArXiv*, abs/2303.16203, 2023.

[22] Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. *ArXiv*, abs/2012.02780, 2020.

[23] Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *International Conference on Machine Learning*, 2023.

[24] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *ArXiv*, abs/2206.00927, 2022.

[25] Taehong Moon, Moonseok Choi, Gayoung Lee, Jung-Woo Ha, Juho Lee, AI Kaist, Naver AI Lab, and Aitrics. Fine-tuning diffusion models with limited data. 2022.

[26] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. *ArXiv*, abs/2002.09797, 2020.

[27] Eddie Y. K. Ng, Feifei Liu, Chengyu Liu, Lina Zhao, X. Zhang, Xiaoling Wu, Xiaoyan Xu, Yulin Liu, Caiyun Ma, Shoushui Wei, Zhiqiang He, and Jianqing Li. An open access database for evaluating the algorithms of electrocardiogram rhythm and morphology abnormality detection. *Journal of Medical Imaging and Health Informatics*, 2018.

[28] Utkarsh Ojha, Yijun Li, Jingwan Lu, Alexei A. Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. Few-shot image generation via cross-domain correspondence. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10738–10747, 2021.

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

[30] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022.

[31] Benjamin Rhodes, Kai Xu, and Michael U. Gutmann. Telescoping density-ratio estimation. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4905–4916. Curran Associates, Inc., 2020.

[32] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22500–22510, 2022.

[33] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L. Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, Seyedeh Sara Mahdavi, Raphael Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *ArXiv*, abs/2205.11487, 2022.

[34] Kunpeng Song, Ligong Han, Bingchen Liu, Dimitris N. Metaxas, and A. Elgammal. Diffusion guided domain adaptation of image generators. *ArXiv*, abs/2212.04473, 2022.

[35] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Conference on Uncertainty in Artificial Intelligence*, 2019.

[36] Yang Song, Jascha Narain Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ArXiv*, abs/2011.13456, 2021.

[37] Nils Strodthoff, Patrick Wagner, Tobias Schaeffter, and Wojciech Samek. Deep learning for ecg analysis: Benchmarks and insights from ptb-xl. *IEEE Journal of Biomedical and Health Informatics*, 25:1519–1528, 2020.

[38] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density Ratio Estimation in Machine Learning*. Cambridge University Press, 2012.

[39] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: Conditional score-based diffusion models for probabilistic time series imputation. *ArXiv*, abs/2107.03502, 2021.

[40] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23:1661–1674, 2011.

[41] Patrick Wagner, Nils Strodthoff, Ralf Bousseljot, Dieter Kreiseler, Fatima I Lunze, Wojciech Samek, and Tobias Schaeffter. Ptb-xl, a large publicly available electrocardiography dataset. *Scientific Data*, 7, 2020.

[42] Chendong Xiang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. A closer look at parameter-efficient tuning in diffusion models. *ArXiv*, abs/2303.18181, 2023.

[43] Jiayu Xiao, Liang Li, Chaofei Wang, Zhengjun Zha, and Qingming Huang. Few shot generative model adaption via relaxed spatial structural alignment. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11194–11203, 2022.

[44] Enze Xie, Lewei Yao, Han Shi, Zhili Liu, Daquan Zhou, Zhaoqiang Liu, Jiawei Li, and Zhenguo Li. Difffit: Unlocking transferability of large diffusion models via simple parameter-efficient fine-tuning. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4207–4216, 2023.

[45] Ceyuan Yang, Yujun Shen, Zhiyi Zhang, Yinghao Xu, Jiapeng Zhu, Zhirong Wu, and Bolei Zhou. One-shot generative domain adaptation. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7699–7708, 2021.

[46] Mengping Yang and Zhe Wang. Image synthesis under limited data: A survey and taxonomy. *ArXiv*, abs/2307.16879, 2023.

[47] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3813–3824, 2023.

[48] Yabo Zhang, Mingshuai Yao, Yuxiang Wei, Zhilong Ji, Jinfeng Bai, and Wangmeng Zuo. Towards diverse and faithful one-shot adaption of generative adversarial networks. *ArXiv*, abs/2207.08736, 2022.

[49] Zicheng Zhang, Yinglu Liu, Congying Han, Tiande Guo, Ting Yao, and Tao Mei. Generalized one-shot domain adaptation of generative adversarial networks. In *Neural Information Processing Systems*, 2022.

[50] Miaoyun Zhao, Yulai Cong, and Lawrence Carin. On leveraging pretrained gans for generation with limited data. In *International Conference on Machine Learning*, 2020.

[51] Yunqing Zhao, Henghui Ding, Houjing Huang, and Ngai-Man Cheung. A closer look at few-shot image generation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9130–9140, 2022.

[52] Jin Zhu, Huimin Ma, Jiansheng Chen, and Jian Yuan. Few-shot image generation with diffusion models. *ArXiv*, abs/2211.03264, 2022.

[53] Jin Zhu, Huimin Ma, Jiansheng Chen, and Jian Yuan. Domainstudio: Fine-tuning diffusion models for domain-driven image generation using limited data. *ArXiv*, abs/2306.14153, 2023.

A More Discussion on Related Work

Finetune diffusion model on limited data Directly finetuning the pre-trained generative model on limited data from the target domain may suffer from overfitting and diversity degradation. Moon et. al. [25] introduce a time-aware adapter inside the attention block. Since the attention modules take about 10% of parameters in the entire diffusion model, they significantly reduced the turning parameters and alleviated the overfitting. While in [44], the authors only finetune specific parameters related to bias, class embedding, normalization, and scale factor. Zhu et. al. [52] found out the images generated by directly finetuned diffusion models share similar features like facial expressions and lack ample high-frequency details. Therefore, they introduce two regularization terms, pairwise similarity loss for diversity and high-frequency components loss to enhance the high-frequency feature.

The main drawback of finetuning the pre-trained diffusion model is the sample complexity is larger compared with training a classifier since modeling the distribution is a tougher task. In our work, we decompose the diffusion model on the target domain as the diffusion model on the source domain plus a guidance network. Since training a guidance network (essential as a classifier demonstrated in section 3) requires smaller sample complexity, we believe this novel framework might provide a new way for diffusion-based domain adaptation.

Text-to-image diffusion model and learning with human feedback Numerous studies on Text-to-Image diffusion models focus on optimizing the diffusion model to align with human preferences and personalize its performance for specific tasks. These endeavors commonly involve strategies such as text-guided zero-shot finetuning [34, 29] or finetuning diffusion model (or its adaptor) through reward-weighted objectives [32, 19, 11, 20, 10]. We acknowledge the significant potential in these approaches, given that language models inherently encapsulate rich semantic information, thereby endowing text-to-image diffusion models with zero-shot transferability. However, it is noteworthy that in domains lacking a substantial amount of paired data for learning semantic mappings, such as biomedical signal processing and electrocardiogram (ECG) data, we refrain from considering these methods as the primary benchmarks in our comparative analysis.

Non-diffusion based approaches in generative domain adaptation Numerous works in generative domain adaptation (or few-shot generative adaptation) study how to improve the transferability of the generative model on limited data from the target domain. Since we mainly focus on the diffusion model, we summarize the primary GAN-based domain adaptation there. They mainly propose to add different kinds of regularization to avoid model collapse [28, 51, 49, 48, 9, 15, 43] or finetune subset of the parameter (adaptor) [1, 45, 22, 50].

B More Discussion on the Potential of the Proposed Method

In this section, we demonstrate the proposed framework is general enough to deal with text-to-image generation tasks and heterogeneous transfer learning.

B.1 Text-to-Image Generation Tasks

Given a source distribution $(x, \boldsymbol{c}_t) \sim p$, where $\boldsymbol{c}_t$ denotes text prompts by using the terminology from [47], a pre-trained diffusion model can be trained on the source distribution. Given a target distribution $(x, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f}) \sim q$, where $\boldsymbol{c}_f$ denotes a task-specific condition, Zhang et al. [47] can fine-tune the pre-trained model by noise matching objective,

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{t}, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f}, \epsilon \sim \mathcal{N}(0,1)} \left[\| \epsilon - \epsilon_\theta \left(\boldsymbol{x}_t, \boldsymbol{t}, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f} \right) \|_2^2 \right].$$

Our method can directly estimate $\nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t, \boldsymbol{c}_t)} \left[\frac{q(\mathbf{x}_0, \boldsymbol{c}_t, \boldsymbol{c}_\mathrm{f})}{p(\mathbf{x}_0, \boldsymbol{c}_t)} \right]$ rather than fine-tuning the pre-trained diffusion model. Domain classifier $c_w(x, y)$ can still be used for estimating the density ratio, where y denotes the embedding of the condition. Moreover, directly fine-tuning the diffusion model on data from the target domain used by [47] is similar to the consistency regularization proposed in our work, while they have a more in-depth design for the architecture and have great results on vision-language tasks. However, with limited data from the target distribution, direct fine-tuning may not achieve good enough performance, which is verified in the two-dimensional Gaussian

setting. In [47], Zhang et al. propose to use zero convolution layers, i.e., 1×1 convolution layer with both weight and bias initialized to zeros, which alleviates the instability of fine-tuning process. This is very different from our methodology which relies on the smaller sample complexity of the classifier/density ratio estimator.

B.2 Heterogeneous Transfer Learning

When the source and target domain contain different class labels, our framework is still applicable, i.e., when $y_t \neq y_s$,

$$\underbrace{\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t, y_t, t)}_{\text{target source}} = \underbrace{\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|y_s)}_{\substack{\text{pre-trained conditional model} \\ \text{on source}}} + \underbrace{\nabla_{\mathbf{x}_t} \log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t, y_s)} \left[\frac{q(\mathbf{x}_0, y_t)}{p(\mathbf{x}_0, y_s)} \right]}_{\text{conditional guidance}}.$$

To generate an unseen class y_t, the key problem here is to choose a particular class from the source domain y_s such that we can borrow useful information from the source domain to generate this unseen class from the target domain. The coupling between y_t and y_s can be learned by solving a static optimal transport problem. More in-depth design, e.g. coupling solved by static optimal transport, can be left to future work.

C Theoretical Details for Section 3

C.1 Proof of Theorem 3.1

Proof. To prove Eq (6), we first build the connection between Score Matching on the target domain and Importance Weighted Denoising Score Matching on the source domain in the following Lemma.

Lemma C.1. *Score Matching on the target domain is equivalent to Importance Weighted Denoising Score Matching on the source domain, i.e.,*

$$\begin{aligned} \boldsymbol{\phi}^* =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q_t(\mathbf{x}_t)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t) \|_2^2 \right] \right\} \\ =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{p(\mathbf{x}_0)} \mathbb{E}_{p(\mathbf{x}_t|\mathbf{x}_0)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0) \|_2^2 \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right] \right\}. \end{aligned} \tag{16}$$

Proof of Lemma C.1. We first connect Score Matching objective in the target domain to Denoising Score Matching objective in target distribution, which is proven by [40], i.e.,

$$\begin{aligned} \boldsymbol{\phi}^* =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q_t(\mathbf{x}_t)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t) \|_2^2 \right] \right\} \\ =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q(\mathbf{x}_0)} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0) \|_2^2 \right] \right\}. \end{aligned}$$

Then, we split the mean squared error of Denoising Score Matching objective on target distribution into three terms as follows:

$$\begin{aligned} & \mathbb{E}_{q(\mathbf{x}_0)} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0) \|_2^2 \right] \\ =& \mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_t)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) \|_2^2 \right] - 2\mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_t)} \left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t), \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0) \rangle \right] + C_1, \end{aligned} \tag{17}$$

where $C_1 = \mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_t)} \left[\| \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0) \|_2^2 \right]$ is a constant independent with $\boldsymbol{\phi}$. We can similarly split the objective function in the right-hand side (RHS) of Eq (16) as follows:

$$\begin{aligned} & \mathbb{E}_{p(\mathbf{x}_0)} \mathbb{E}_{p(\mathbf{x}_t|\mathbf{x}_0)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0) \|_2^2 \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right] \\ =& \mathbb{E}_{p(\mathbf{x}_0, \mathbf{x}_t)} \left[\| \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t) \|_2^2 \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right] - 2\mathbb{E}_{p(\mathbf{x}_0, \mathbf{x}_t)} \left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, t), \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0) \rangle \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)} \right] + C_2, \end{aligned} \tag{18}$$

where C_2 is a constant independent with $\boldsymbol{\phi}$. It is easy to show that the first term in Eq (17) is equal to the first term in Eq (18), i.e.,

$$\begin{aligned}
\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t)}\left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t)\|_2^2\,\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right] &= \int_{\mathbf{x}_0}\int_{\mathbf{x}_t} p(\mathbf{x}_0)p(\mathbf{x}_t|\mathbf{x}_0)\,\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t)\|_2^2\,\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0 d\mathbf{x}_t \\
&\overset{(i)}{=} \int_{\mathbf{x}_0}\int_{\mathbf{x}_t} p(\mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_0)\,\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t)\|_2^2\,\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0 d\mathbf{x}_t \\
&= \int_{\mathbf{x}_0}\int_{\mathbf{x}_t} q(\mathbf{x}_0,\mathbf{x}_t)\,\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t)\|_2^2\,d\mathbf{x}_0 d\mathbf{x}_t \\
&= \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t)}\left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t)\|_2^2\right],
\end{aligned}$$

where the equality (i) is due to $q(\mathbf{x}_t|\mathbf{x}_0) = p(\mathbf{x}_t|\mathbf{x}_0)$.

Next, we prove the second terms in Eq (17) and Eq (18) are also equivalent:

$$\begin{aligned}
&\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t)}\left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t), \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\rangle \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right] \\
=&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t} p(\mathbf{x}_0,\mathbf{x}_t)\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t), \frac{\nabla_{\mathbf{x}_t} p(\mathbf{x}_t|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)}\rangle \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0 d\mathbf{x}_t \\
\overset{(i)}{=}&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t} p(\mathbf{x}_0)p(\mathbf{x}_t|\mathbf{x}_0)\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t), \frac{\nabla_{\mathbf{x}_t} q(\mathbf{x}_t|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)}\rangle \frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0 d\mathbf{x}_t \\
=&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t} \langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t), \nabla_{\mathbf{x}_t} q(\mathbf{x}_t|\mathbf{x}_0)\rangle q(\mathbf{x}_0)d\mathbf{x}_0 d\mathbf{x}_t \\
=&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t} \langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t), \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0)\rangle q(\mathbf{x}_t|\mathbf{x}_0)q(\mathbf{x}_0)d\mathbf{x}_0 d\mathbf{x}_t \\
=&\mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t)}\left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t), \nabla_{\mathbf{x}_t}\log q(\mathbf{x}_t|\mathbf{x}_0)\rangle\right],
\end{aligned}$$

where the equality (i) is again due to $q(\mathbf{x}_t|\mathbf{x}_0) = p(\mathbf{x}_t|\mathbf{x}_0)$. Thereby we prove Eq 16. □

According to Lemma C.1,

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t\left\{\lambda(t)\mathbb{E}_{p(\mathbf{x}_0)}\mathbb{E}_{p(\mathbf{x}_t|\mathbf{x}_0)}\left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,t) - \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\|_2^2\,\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]\right\}.$$

Based on this, we may use Importance Weighted Denoising Score Matching on the source domain to get the analytic form of $\mathbf{s}_{\boldsymbol{\phi}^*}$ as follows:

$$\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t,t) = \frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]}.$$

The RHS of Eq (6) can be rewritten as follows:

$$\begin{aligned}
\text{RHS} =&\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t}\log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right] = \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) + \frac{\nabla_{\mathbf{x}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]} \\
=&\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) + \frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_0|\mathbf{x}_t)\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]}.
\end{aligned}$$

Since

$$\begin{aligned}
\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_0|\mathbf{x}_t) &= \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0) + \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_0) - \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t) \\
&= \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0) - \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t),
\end{aligned}$$

we can further rewrite the RHS of Eq (6) as follows:

$$\begin{aligned}\text{RHS} =&\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)+\frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]}-\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)\\ =&\frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]}\\ =&\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t,t).\end{aligned}$$

Thereby we complete the proof. □

C.2 Proof of Lemma 3.2

Proof. The proof is straightforward and we include it below for completeness. Note that the objective function can be rewritten as

$$\begin{aligned}\mathcal{L}_{\text{guidance}}(\boldsymbol{\psi}):=&\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t)}\left[\left\|h_{\boldsymbol{\psi}}\left(\mathbf{x}_t,t\right)-\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right\|_2^2\right]\\ =&\int_{\mathbf{x}_t}\left\{\int_{\mathbf{x}_0}p(\mathbf{x}_0|\mathbf{x}_t)\left\|h_{\boldsymbol{\psi}}\left(\mathbf{x}_t,t\right)-\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right\|_2^2 d\mathbf{x}_0\right\}p(\mathbf{x}_t)d\mathbf{x}_t\\ =&\int_{\mathbf{x}_t}\left\{\|h_{\boldsymbol{\psi}}(\mathbf{x}_t,t)\|_2^2-2\langle h_{\boldsymbol{\psi}}(\mathbf{x}_t,t),\int_{\mathbf{x}_0}p(\mathbf{x}_0|\mathbf{x}_t)\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0\rangle\right\}p(\mathbf{x}_t)d\mathbf{x}_t+C\\ =&\int_{\mathbf{x}_t}\left\|h_{\boldsymbol{\psi}}(\mathbf{x}_t,t)-\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right]\right\|_2^2 p(\mathbf{x}_t)d\mathbf{x}_t,\end{aligned}$$

where C is a constant independent of $\boldsymbol{\psi}$. Thus we have the minimizer $\boldsymbol{\psi}^*=\arg\min_{\boldsymbol{\psi}}\mathcal{L}_{\text{guidance}}(\boldsymbol{\psi})$ satisfies $h_{\boldsymbol{\psi}^*}\left(\mathbf{x}_t,t\right)=\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[q(\mathbf{x}_0)/p(\mathbf{x}_0)\right]$. □

C.3 Conditional version for Lemma 3.2

Lemma C.2. *For a neural network $h_{\boldsymbol{\psi}}\left(\mathbf{x}_t,y,t\right)$ parameterized by $\boldsymbol{\psi}$, define the objective*

$$\mathcal{L}_{guidance}(\boldsymbol{\psi}):=\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t,y)}\left[\left\|h_{\boldsymbol{\psi}}\left(\mathbf{x}_t,y,t\right)-\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right\|_2^2\right],\tag{19}$$

then its minimizer $\boldsymbol{\psi}^*=\arg\min_{\boldsymbol{\psi}}\mathcal{L}_{guidance}(\boldsymbol{\psi})$ satisfies:

$$h_{\boldsymbol{\psi}^*}\left(\mathbf{x}_t,y,t\right)=\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[q(\mathbf{x}_0,y)/p(\mathbf{x}_0,y)\right].$$

Proof of Lemma C.2

Proof. The proof is straightforward and we include it below for completeness. Note that the objective function can be rewritten as

$$\begin{aligned}&\mathcal{L}_{\text{guidance}}(\boldsymbol{\psi})\\ :=&\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t,y)}\left[\left\|h_{\boldsymbol{\psi}}\left(\mathbf{x}_t,y,t\right)-\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right\|_2^2\right]\\ =&\int_{\mathbf{x}_t}\int_y\left\{\int_{\mathbf{x}_0}p(\mathbf{x}_0|\mathbf{x}_t,y)\left\|h_{\boldsymbol{\psi}}\left(\mathbf{x}_t,y,t\right)-\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right\|_2^2 d\mathbf{x}_0\right\}p(\mathbf{x}_t|y)p(y)dyd\mathbf{x}_t\\ =&\int_{\mathbf{x}_t}\int_y\left\{\|h_{\boldsymbol{\psi}}(\mathbf{x}_t,y,t)\|_2^2-2\langle h_{\boldsymbol{\psi}}(\mathbf{x}_t,y,t),\int_{\mathbf{x}_0}p(\mathbf{x}_0|\mathbf{x}_t,y)\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}d\mathbf{x}_0\rangle\right\}p(\mathbf{x}_t|y)p(y)dyd\mathbf{x}_t+C\\ =&\int_{\mathbf{x}_t}\int_y\left\|h_{\boldsymbol{\psi}}(\mathbf{x}_t,y,t)-\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]\right\|_2^2 p(\mathbf{x}_t|y)p(y)dyd\mathbf{x}_t,\end{aligned}$$

where C is a constant independent of $\boldsymbol{\psi}$. Thus we have the minimizer $\boldsymbol{\psi}^* = \arg\min_{\boldsymbol{\psi}} \mathcal{L}_{\text{guidance}}(\boldsymbol{\psi})$ satisfies $h_{\boldsymbol{\psi}^*}(\mathbf{x}_t, y, t) = \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}[q(\mathbf{x}_0, y)/p(\mathbf{x}_0, y)]$. □

C.4 Proof of Theorem 3.3

Proof. The proof is similar to the proof of Theorem 3.1. To prove Eq 11, we first build the connection between the Conditional Score Matching on the target domain and Importance Weighted Conditional Denoising Score Matching on the source domain in the following Lemma:

Lemma C.3. *Conditional Score Matching on the target domain is equivalent to Importance Weighted Denoising Score Matching on the source domain, i.e.,*

$$\begin{aligned}\boldsymbol{\phi}^* =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q_t(\mathbf{x}_t,y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t|y)\|_2^2 \right] \right\} \\ =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{p(\mathbf{x}_0,y)} \mathbb{E}_{p(\mathbf{x}_t|\mathbf{x}_0)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0)\|_2^2 \frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} \right] \right\}.\end{aligned}$$

Proof of Lemma C.3. We first connect the Conditional Score Matching objective in the target domain to the Conditional Denoising Score Matching objective in target distribution, which is proven by [4, Theorem 1], i.e.,

$$\begin{aligned}\boldsymbol{\phi}^* =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q_t(\mathbf{x}_t,y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log q_t(\mathbf{x}_t|y)\|_2^2 \right] \right\} \\ =& \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{q(\mathbf{x}_0,y)} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0)\|_2^2 \right] \right\}.\end{aligned}$$

Then we split the mean squared error of the Conditional Denoising Score Matching objective on target distribution into three terms as follows:

$$\begin{aligned}&\mathbb{E}_{q(\mathbf{x}_0,y)} \mathbb{E}_{q(\mathbf{x}_t|\mathbf{x}_0)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0)\|_2^2 \right] \\ =& \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t,y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t)\|_2^2 \right] - 2\mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t,y)} \left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t), \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0) \rangle \right] + C_1, \end{aligned} \tag{20}$$

where $C_1 = \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t,y)} \left[\|\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0)\|_2^2 \right]$ is a constant independent with $\boldsymbol{\phi}$, and $q(\mathbf{x}_t|\mathbf{x}_0, y) = q(\mathbf{x}_t|\mathbf{x}_0)$ because of conditional independent of $\mathbf{x}_t$ and y given $\mathbf{x}_0$ by assumption. We can similarly split the mean squared error of Denoising Score Matching on the source domain into three terms as follows:

$$\begin{aligned}&\mathbb{E}_{p(\mathbf{x}_0,y)} \mathbb{E}_{p(\mathbf{x}_t|\mathbf{x}_0)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t) - \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0)\|_2^2 \frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} \right] \\ =& \mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t,y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t)\|_2^2 \frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} \right] - 2\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t,y)} \left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t), \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t|\mathbf{x}_0) \rangle \frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} \right] \\ &+ C_2, \end{aligned} \tag{21}$$

where C_2 is a constant independent with $\boldsymbol{\phi}$.

It is obvious to show that the first term in Eq (20) is equal to the first term in Eq (21), i.e.,

$$\begin{aligned}&\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t,y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t)\|_2^2 \frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} \right] \\ =& \int_{\mathbf{x}_0} \int_{\mathbf{x}_t} \int_y p(\mathbf{x}_0, y) p(\mathbf{x}_t|\mathbf{x}_0) \|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t)\|_2^2 \frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} d\mathbf{x}_0 d\mathbf{x}_t dy \\ =& \int_{\mathbf{x}_0} \int_{\mathbf{x}_t} \int_y p(\mathbf{x}_0, y) q(\mathbf{x}_t|\mathbf{x}_0) \|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t)\|_2^2 \frac{q(\mathbf{x}_0, y)}{p(\mathbf{x}_0, y)} d\mathbf{x}_0 d\mathbf{x}_t dy \\ =& \int_{\mathbf{x}_0} \int_{\mathbf{x}_t} \int_y q(\mathbf{x}_0, \mathbf{x}_t, y) \|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t)\|_2^2 d\mathbf{x}_0 d\mathbf{x}_t dy \\ =& \mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t,y)} \left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t, y, t)\|_2^2 \right].\end{aligned}$$

And the second term is also equivalent:

$$
\begin{aligned}
&\mathbb{E}_{p(\mathbf{x}_0,\mathbf{x}_t,y)}\left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,y,t), \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\rangle \frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right] \\
=&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t}\int_y p(\mathbf{x}_0,\mathbf{x}_t,y)\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,y,t), \frac{\nabla_{\mathbf{x}_t} p(\mathbf{x}_t|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)}\rangle \frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)} d\mathbf{x}_0 d\mathbf{x}_t dy \\
=&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t}\int_y p(\mathbf{x}_0,\mathbf{x}_t,y)\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,y,t), \frac{\nabla_{\mathbf{x}_t} q(\mathbf{x}_t|\mathbf{x}_0)}{p(\mathbf{x}_t|\mathbf{x}_0)}\rangle \frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)} d\mathbf{x}_0 d\mathbf{x}_t dy \\
=&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t}\int_y \langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,y,t), \nabla_{\mathbf{x}_t} q(\mathbf{x}_t|\mathbf{x}_0)\rangle q(\mathbf{x}_0,y) d\mathbf{x}_0 d\mathbf{x}_t dy \\
=&\int_{\mathbf{x}_0}\int_{\mathbf{x}_t}\int_y \langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,y,t), \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t|\mathbf{x}_0)\rangle q(\mathbf{x}_t|\mathbf{x}_0) q(\mathbf{x}_0,y) d\mathbf{x}_0 d\mathbf{x}_t dy \\
=&\mathbb{E}_{q(\mathbf{x}_0,\mathbf{x}_t,y)}\left[\langle \mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,y,t), \nabla_{\mathbf{x}_t}\log q(\mathbf{x}_t|\mathbf{x}_0)\rangle\right].
\end{aligned}
$$

□

According to Lemma C.3, the optimal solution satisfies

$$
\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \mathbb{E}_t\left\{\lambda(t)\mathbb{E}_{p(\mathbf{x}_0,y)}\mathbb{E}_{p(\mathbf{x}_t|\mathbf{x}_0)}\left[\|\mathbf{s}_{\boldsymbol{\phi}}(\mathbf{x}_t,y,t) - \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\|_2^2 \frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]\right\},
$$

Then, we use Importance Weighted Conditional Denoising Score Matching on the source domain to get the analytic form of $\mathbf{s}_{\boldsymbol{\phi}^*}$ as follows:

$$
\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t,y,t) = \frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]}.
$$

Moreover, the RHS of Eq (11) can be rewritten as:

$$
\begin{aligned}
\text{RHS} =&\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y) + \nabla_{\mathbf{x}_t}\log \mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right] \\
=&\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y) + \frac{\nabla_{\mathbf{x}_t}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]} \\
=&\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y) + \frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_0|\mathbf{x}_t,y)\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]}.
\end{aligned}
$$

Since

$$
\begin{aligned}
\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_0|\mathbf{x}_t,y) &= \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0,y) + \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_0|y) - \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y) \\
&= \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0,y) - \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y), \\
&= \nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0) - \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y),
\end{aligned}
$$

we can further simplify the RHS of Eq (11) as follows:

$$
\begin{aligned}
\text{RHS} =&\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y) + \frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]} - \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t|y) \\
=&\frac{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\nabla_{\mathbf{x}_t}\log p(\mathbf{x}_t|\mathbf{x}_0)\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]}{\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t,y)}\left[\frac{q(\mathbf{x}_0,y)}{p(\mathbf{x}_0,y)}\right]} \\
=&\mathbf{s}_{\boldsymbol{\phi}^*}(\mathbf{x}_t,t).
\end{aligned}
$$

Thereby, we finish the proof. □

C.5 Proof for Cycle Regularization

Proof of Eq (12).

$$\begin{aligned}\mathbb{E}_{p(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}\right] &= \int p(\mathbf{x}_0|\mathbf{x}_t)\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0 = \int \frac{p(\mathbf{x}_t|\mathbf{x}_0)p(\mathbf{x}_0)}{p_t(\mathbf{x}_t)}\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0 \\ &= \int \frac{q(\mathbf{x}_t|\mathbf{x}_0)p(\mathbf{x}_0)}{p_t(\mathbf{x}_t)}\frac{q(\mathbf{x}_0)}{p(\mathbf{x}_0)}d\mathbf{x}_0 = \int q(\mathbf{x}_t|\mathbf{x}_0)\frac{q(\mathbf{x}_0)}{p_t(\mathbf{x}_t)}d\mathbf{x}_0 \\ &= \int \frac{q(\mathbf{x}_0|\mathbf{x}_t)q_t(\mathbf{x}_t)}{q(\mathbf{x}_0)}\frac{q(\mathbf{x}_0)}{p_t(\mathbf{x}_t)}d\mathbf{x}_0 = \int q(\mathbf{x}_0|\mathbf{x}_t)\frac{q_t(\mathbf{x}_t)}{p_t(\mathbf{x}_t)}d\mathbf{x}_0 \\ &= \mathbb{E}_{q(\mathbf{x}_0|\mathbf{x}_t)}\left[\frac{q_t(\mathbf{x}_t)}{p_t(\mathbf{x}_t)}\right],\end{aligned}$$

where $p_t(\mathbf{x}_t) = \int p(\mathbf{x}_0)p(\mathbf{x}_t|\mathbf{x}_0)d\mathbf{x}_0$ and $q_t(\mathbf{x}_t) = \int q(\mathbf{x}_0)q(\mathbf{x}_t|\mathbf{x}_0)d\mathbf{x}_0$ are the marginal distributions at time t of source and target distributions, respectively. □

D More Details on Experiments

D.1 Algorithms for TGDP

TGDP adopts Algorithm 1 and 2 for training a domain classifier and Algorithm 3 and 4 for training the guidance network.

Algorithm 1 Training a domain classifier

Require: Samples from the marginal distribution of the source domain $p(\mathbf{x})$ and target domain $q(\mathbf{x})$, and initial weights of domain classifier $\boldsymbol{\omega}$.

1: **repeat**
2: Sample mini-batch data from source distribution and target distribution respectively with batch size b.
3: Take gradient descent step on

$$\nabla_{\boldsymbol{\omega}}\left\{-\frac{1}{b}\sum_{\mathbf{x}_i\in p}[\log c_{\boldsymbol{\omega}}(\mathbf{x}_i)] - \frac{1}{b}\sum_{\mathbf{x}'_i\in q}[\log(1-c_{\boldsymbol{\omega}}(\mathbf{x}'_i))]\right\}.$$

4: **until** converged.
5: **return** weights of domain classifier $\boldsymbol{\omega}$.

Algorithm 2 Training a time-dependent domain classifier

Require: Samples from the marginal distribution of the source domain $p(\mathbf{x})$ and target domain $q(\mathbf{x})$, pre-defined forward transition $p(\mathbf{x}_t|\mathbf{x}_0)$, and initial weights of domain classifier $\boldsymbol{\omega}$.

1: **repeat**
2: Sample mini-batch data from source distribution and target distribution respectively with batch size b.
3: Sample time $t \sim \mathrm{Uniform}(\{1,\dots,T\})$ and perturb $\mathbf{x}_0$ by forward transition $p(\mathbf{x}_t|\mathbf{x}_0)$.
4: Take gradient descent step on

$$\nabla_{\boldsymbol{\omega}}\left\{-\frac{1}{b}\sum_{\mathbf{x}_0\sim p}\sum_{\mathbf{x}_t|\mathbf{x}_0}[\log c_{\boldsymbol{\omega}}(\mathbf{x}_t,t)] - \frac{1}{b}\sum_{\mathbf{x}_0\sim q}\sum_{\mathbf{x}_t|\mathbf{x}_0}[\log(1-c_{\boldsymbol{\omega}}(\mathbf{x}_t,t))]\right\}.$$

5: **until** converged.
6: **return** weights of time-dependent domain classifier $\boldsymbol{\omega}$.

Algorithm 3 Training a guidance network (without regularization)

Require: Samples from the marginal distribution of the source domain $p(\mathbf{x})$, pre-defined forward transition $p(\mathbf{x}_t|\mathbf{x}_0)$, pre-trained domain classifier $c_{\boldsymbol{\omega}}$, and initial weights of guidance network $\boldsymbol{\psi}$.

1: **repeat**

2: Sample mini-batch data from source distribution $\mathbf{x}_0$ with batch size b.

3: Sample time $t \sim \text{Uniform}(\{1, \ldots, T\})$ and perturb $\mathbf{x}_0$ by forward transition $p(\mathbf{x}_t|\mathbf{x}_0)$.

4: Take gradient descent step on

$$\nabla_{\boldsymbol{\psi}} \left\{ \frac{1}{b} \sum_{\mathbf{x}_0, \mathbf{x}_t} \left[\left\| h_{\boldsymbol{\psi}} \left(\mathbf{x}_t, t \right) - \left(1 - c_{\boldsymbol{\omega}}(\mathbf{x}_0) \right) / c_{\boldsymbol{\omega}}(\mathbf{x}_0) \right\|_2^2 \right] \right\}.$$

5: **until** converged.

6: **return** weights of guidance network $\boldsymbol{\psi}$.

Algorithm 4 Training a guidance network (with regularization)

Require: Samples from the marginal distribution of the source domain $p(\mathbf{x})$ and target domain $q(\mathbf{x})$, pre-trained diffusion model on source distribution $s_{\text{source}}(\mathbf{x}_t, t)$, pre-defined forward transition $q(\mathbf{x}_t|\mathbf{x}_0), p(\mathbf{x}_t|\mathbf{x}_0)$, pre-trained domain classifier $c_{\boldsymbol{\omega}}(\mathbf{x}_0)$ and time dependent domain classifier $c'_{\boldsymbol{\omega}}(\mathbf{x}_0, t)$, hyperparameter η_1, η_2, and initial weights of guidance network $\boldsymbol{\psi}$.

1: **repeat**

2: Sample mini-batch data from source distribution $\mathbf{x}_0$ with batch size b.

3: Perturb $\mathbf{x}_0$ by forward transition $p(\mathbf{x}_t|\mathbf{x}_0)$.

4: $\mathcal{L}_{\text{guidance}}(\boldsymbol{\psi}) = \frac{1}{b} \sum_{\mathbf{x}_0, \mathbf{x}_t, t} \left[\left\| h_{\boldsymbol{\psi}} \left(\mathbf{x}_t, t \right) - \left(1 - c_{\boldsymbol{\omega}}(\mathbf{x}_0) \right) / c_{\boldsymbol{\omega}}(\mathbf{x}_0) \right\|_2^2 \right]$

5: Sample mini-batch data from target distribution $\mathbf{x}'_0$ with batch size b.

6: Sample time $t \sim \text{Uniform}(\{1, \ldots, T\})$ and perturb $\mathbf{x}'_0$ by forward transition $q(\mathbf{x}'_t|\mathbf{x}'_0)$.

7: $\mathcal{L}_{\text{cycle}}(\boldsymbol{\psi}) = \frac{1}{b} \sum_{\mathbf{x}'_0, \mathbf{x}'_t, t} \left[\left\| h_{\boldsymbol{\psi}} \left(\mathbf{x}'_t, t \right) - \left(1 - c'_{\boldsymbol{\omega}}(\mathbf{x}'_0, t) \right) / c'_{\boldsymbol{\omega}}(\mathbf{x}'_0, t) \right\|_2^2 \right].$

8: $\mathcal{L}_{\text{consistence}}(\boldsymbol{\psi}) = \frac{1}{b} \sum_{\mathbf{x}'_0, \mathbf{x}'_t, t} \left[\left\| s_{\text{source}}(\mathbf{x}'_t, t) + \nabla_{\mathbf{x}'_t} \log h_{\boldsymbol{\psi}} \left(\mathbf{x}'_t, t \right) - \nabla_{\mathbf{x}'_t} \log q(\mathbf{x}_t|\mathbf{x}'_0) \right\|_2^2 \right].$

9: Take gradient descent step on

$$\nabla_{\boldsymbol{\psi}} \left\{ \mathcal{L}_{\text{guidance}} + \eta_1 \, \mathcal{L}_{\text{cycle}} + \eta_2 \, \mathcal{L}_{\text{consistence}} \right\}.$$

10: **until** converged.

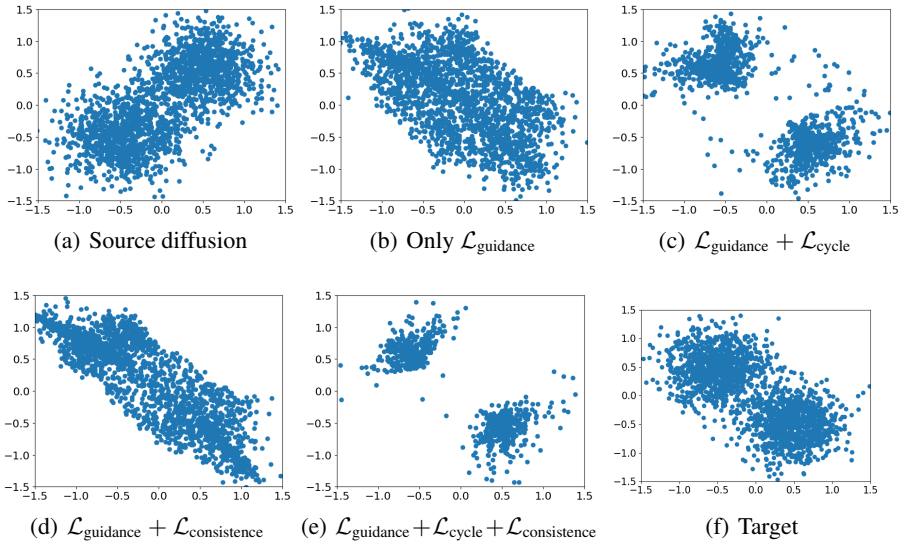
11: **return** weights of guidance network $\boldsymbol{\psi}$.

D.2 Ablation Studies on simulations

In Figure D.2, we demonstrate the ablation studies on simulations. We can see that only using the consistency regularization term (Figure D.2 (b)) is not able to recover the true distribution in the target domain. Our guidance loss together with cycle regularization can learn a good approximation of target distribution while adding consistency regularization can achieve better performance.

D.3 Implementation details for ECG Benchmark

For TGDP and all of the baseline methods, we utilize the same architecture as the conditional generative models for ECG data, SSSM-ECG [2]. For *Vanilla Diffusion*, we train the diffusion model for 100k iterations by Adam optimizer with a learning rate 2e^{-4}. For *Finetune Generator*, we finetune the pre-trained diffusion model for 50k iterations by Adam optimizer with a learning rate 2e^{-5}. For TGDP, we adopt a 4-layer MLP with 512 hidden units and SiLU activation function as the backbone of the guidance network. We train the guidance network for 50k iterations by Adam optimizer with a learning rate 2e^{-4}. For utility evaluation, we adopt the same architecture, xresnet1d50 [37], as the backbone. We train the classifier from sketch for 50 epochs with with a learning rate 1e-2. For Finetune Classifier, we finetune a pre-trained classifier for 30 epochs with with a learning rate 1e-3.



(a) Source diffusion (b) Only $\mathcal{L}_{\text{guidance}}$ (c) $\mathcal{L}_{\text{guidance}} + \mathcal{L}_{\text{cycle}}$

(d) $\mathcal{L}_{\text{guidance}} + \mathcal{L}_{\text{consistence}}$ (e) $\mathcal{L}_{\text{guidance}} + \mathcal{L}_{\text{cycle}} + \mathcal{L}_{\text{consistence}}$ (f) Target

Figure 4: An illustration of the effectiveness of cycle regularization and consistency regularization proposed in Section 3.4.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   [Yes] .

   Justification: We summarize the contributions and scope in Abstract as "we prove the optimal diffusion model for the target domain integrates pre-trained diffusion models with additional guidance" and we also summarize the main contribution of our paper in Introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] .

   Justification: We summarize the limitations of this paper in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes] .

   Justification: We carefully state the assumptions in the statement of all theorems in this paper and all of the theorems are proven in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes] .

   Justification: We provide all of the details to reproduce the experiments in this paper and we also release our code.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same

dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes] .

   Justification: We provide the code for all of the experiments together with clear instructions.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes] .

   Justification: Experimental Setting/Details can be found in our paper and code.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes] .

   Justification: The standard deviation of the experiments in ECG Benchmark has been provided in Table 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes] .

   Justification: We utilized a computing cluster equipped with 6 NVIDIA GeForce 3090 GPUs with memory 24268MiB and Intel(R) Xeon(R) Platinum 8352Y CPUs @ 2.20GHz. The computational costs for each of the individual experimental runs can be found in the following table. The total computational time of all experiments in this paper is around 200 GPU hours.

Experiments	Memory-Usage	Running Time
Experiments in Table 1	1281MiB	5min
Training Vanilla Diffusion	19815MiB	1h
Finetune Generator	19815MiB	40min
Training TGDP	19597MiB	40min
Sampling	9535MiB	21h
Experiments in Table 3	6075MB	10min

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes] .

   Justification: We review the NeurIPS Code of Ethics and ensure our compliance with its requirements.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes] .

    Justification: We summarize the potential negative societal impacts of this paper together with the corresponding solutions to mitigate the negative impacts in Section 5.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA] .

Justification: The main contribution of our paper is an efficient way for transferring a pre-trained model on target distribution. We do not release data or models that have a high risk for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes] .

Justification: The license of the dataset and code used in this paper can be found in the following table.

Assets	License	Link
PTB-XL	CC-BY 4.0	https://physionet.org/content/ptb-xl/1.0.3/
ICBEB2018	CC0: Public Domain	https://www.kaggle.com/datasets/bjoernjostein/china-12lead-ecg-challenge-database
SSSD-ECG	MIT License	https://github.com/AI4HealthUOL/SSSD-ECG?tab=readme-ov-file
ECG Benchmarks	CC-BY 4.0	https://github.com/helme/ecg_ptbxl_benchmarking

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes] .

Justification: The code provided in the supplementary material follows the CC-BY 4.0 license.

Guidelines:

- The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA] .

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA] .

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.