Learning Top-Down Generative Models by Short-Run Markov Chain Monte Carlo Inference with Optimal Transport Correction

Dongsheng An, Jianwen Xie, Ping Li

Abstract—Learning top-down generative models with latent variables via maximum likelihood typically requires to infer latent variables for each training example based on the posterior distribution of latent variables. The inference step relies on either running a time-consuming long-run Markov chain Monte Carlo (MCMC) sampling from the posterior distribution or constructing a separate inference model for variational learning. In this paper, we propose to use an efficient short-run MCMC, such as Langevin dynamics, as an approximate inference engine. The bias in the aggregated posterior distribution of the inferred latent variables obtained by the non-convergent short-run MCMC is corrected by optimal transport (OT), which aims at transforming the biased distribution to the prior distribution with minimum transport cost. The proposed algorithm alternates the short-run MCMC inference step, the OT correction step, and the parameter learning step. In each iteration, with more reliable latent variables obtained from the inference step and the correction step, the parameter learning step can be more accurate. Our experiments not only verify the effectiveness of the OT correction for the short-run MCMC inference, but also demonstrate that the generative models trained by the proposed strategy outperform the variational auto-encoder and the MCMC inference without using OT correction in the tasks of image reconstruction, image generation, image inpainting, anomaly detection and unsupervised image recovery.

Index Terms—Deep generative models, Top-down generator, Latent variable model, Short-run MCMC, Langevin dynamics, Optimal transport, image synthesis, image recovery, unsupervised learning

1 Introduction

Generative model is a powerful tool to learn any kind of data distribution in an unsupervised manner. The goal of all types of generative models is to estimate the true data distribution from some observed training data so as to generate novel and realistic data examples. Training generative models is not only a fundamental problem in statistics and machine learning, but also important to unsupervised and semi-supervised learning applications in artificial intelligence and computer vision. In general, generative models can be categorized into two classes, which differ in the following aspects: (i) *Explicit density v.s. implicit density*: A generative model can represent a data distribution either explicitly or implicitly. From the model perspective, the generative model with an explicit probability density of the data is typically in the form of energy-based model [1] or Markov random field model [2], while the generative model with an implicit density is in the form of latent variable model, which typically assumes the data to be generated from some latent variables that follow a prior distribution, so that the probability density of the data is implicit because it can be obtained by integrating out the latent variables but this integral is analytically intractable. (ii) *implicit generation v.s. explicit generation*: All generative models have potentials to generate data from the learned explicit or implicit distributions. From the generation perspective, the "explicit" model generates data via an implicit way, in which an iterative Markov chain Monte Carlo (MCMC) [3], [4] sampling process is required to explore and discover local modes in the learned distribution, while the "implicit" model generates data in a more explicit way where it first samples the latent variables from the prior distribution and then transforms them to the data. This is called ancestral sampling. (iii) *bottom-up descriptive v.s. top-down generative*: According to the terminology of [5], the "explicit" generative model is also called the descriptor model, since the probability density is built on descriptive feature statistics computed from the input data by a bottom-up process. The "implicit" generative model is defined by a top-down structure, which directly maps the latent variables to the data. Such a model is also called the generator model following [6]. In this paper, we only study the top-down generative model, which is of the form of the latent variable model that lacks a close form of probability distribution of the data but is easy to generate data.

Thanks to the powerful approximation ability of deep neural networks, recent years have seen a great success of deep top-down generative models in numerous computer vision and machine learning applications, such as image generation and synthesis [6], [7], [8], [9], [10], [11], [12], image recovery [13], [14], [15], [16], [17], image representation [18], [19], image disentanglement [20], [21], [22], anomaly detection [23], [24], zero-shot learning [25], [26], salient object prediction [27], [28], video generation [29], [30], [31], [32] etc. Such models typically include simple but expressive deep generator networks, which generate each observed example from a low-dimensional vector of latent variables, and the latent vector is assumed to follow a non-informative prior distribution, such as Gaussian white noise distribution. Since high-dimensional visual data (e.g., images and videos) usually lie on low-dimensional manifolds, learning latent variable models of visual data is of fundamental importance in the field of computer vision for the sake of unsupervised feature learning and disentangled representation learning. In the likelihood-based training of the

• *D. An, J. Xie, and P. Li are with Cognitive Computing Lab, Baidu Research, 10900 NE 8th St. Bellevue, WA 98004, USA. E-mail: dongshengan15@gmail.com, jianwen.kenny@gmail.com, pingli98@gmail.com.*

Manuscript received xxx, 2021.

top-down generative models, the challenge mainly comes from the inference of the latent variables for each observation, which typically relies on MCMC methods to draw fair samples from the analytically intractable posterior distribution (i.e., the conditional distribution of the latent variables given the observed example). Since the top-down generative model is parameterized by a highly non-linear deep neural network, the derived posterior distribution of the latent variables is also parameterized by the deep neural network and is highly intractable. Therefore, the MCMC inference is very challenging and may suffer from the non-convergence and inefficiency problems, which might further affect the accuracy of the subsequent model parameter estimation.

To avoid the inefficient MCMC sampling from the posterior distribution, the variational inference, such as the variational auto-encoder (VAE) [9], becomes an attractive alternative by approximating the intractable posterior distribution with a tractable network. This amortized inference requires an additional minimization of the Kullback-Leibler (KL) divergence between the posterior and the approximating inference network. Despite the growing prevalence and popularity of the VAE, its drawbacks are increasingly obvious and can not be neglected. (i) In variational inference, the model parameterizes the intrinsic iterative inference process by an extrinsic feedforward inference model. These extra parameters in the inference model due to the reparameterization have to be estimated together with those of the main generator network. This may distinctly increase not only the model size in terms of the number of trainable parameters but also the training difficulty. (ii) The joint training of the generator and the inference model in the VAE is to be accomplished by maximizing the variational lower bound. Thus, the accuracy of VAE heavily depends on the accuracy of the inference model as an approximation of the true posterior distribution. Theoretically, only when the KL divergence between the inference model and the posterior distribution is minimized to zero, the variational inference is equivalent to the desired maximum likelihood estimation. However, this goal is usually infeasible in practice because of the limited capacity of the designed inference model and the suboptimal solution obtained in the optimization process. (iii) An extra effort is required to made in designing the inference model of VAE, especially for the top-down generators that have complicated dependency structures with the latent variables, e.g., Nijkamp et al. [12] proposed a top-down generator with multiple layers of latent variables and Xie et al. [30], [31] proposed dynamic generators with time sequences of latent variables. It is not a simple task to design approximating inference models that infer latent variables for models mentioned above. An arbitrary design of the inference model cannot guarantee the performance of the VAE. The VAE model may often suffer from oversimplified posterior approximations.

In this paper, we will totally abandon the idea of reparameterizing the inference process, and will reuse the MCMC inference for training top-down generative models (i.e., deep latent variable models). The reasons why we stick to MCMC inference are that: (i) There has been a great progress of studying variational inference and VAEs recently, but the advance of the MCMC inference for likelihood-based learning of generative models is still lagging behind. We aims at pushing forward the MCMC inference, given that it has so many appealing advantages; (ii) The recent advances of the maximum likelihood estimation (MLE) of generative models with MCMC inference, such as [11], [12], [24], [30], have demonstrated the potential of the MCMC inference. We are encouraged to further investigate and improve the training of top-down generative models using MCMC inference.

To be specific, we study using a short-run MCMC, such as a short-run Langevin dynamics [33], [34], to perform the inference of the latent vectors during model training. The short-run MCMC is always initialized from the same distribution, such as the Gaussian noise distribution, and performed with the same number of Langevin steps. However, the convergence of finite-step Langevin dynamics in each iteration might be questionable, so we accept the bias existing in such a short-run MCMC inference and propose to use the optimal transport (OT) method [35] to correct the bias. The OT can be adopted to transform an arbitrary probability distribution to a desired distribution with a minimum transport cost. Thus, we can use the OT cost to measure the difference between two probability distributions. We treat the short-run MCMC as a learned flow model whose parameters are from the top-down generative model. Even though the short-run MCMC is toward the posterior distribution, its actual output marginal distribution might not exactly follow the assumed prior distribution. In this paper, we not only validate the above phenomenon but also propose to correct the bias of the short-run MCMC by performing an optimal transport from the resulting distribution produced by the short-run MCMC to the target prior distribution. This operation is to minimize the OT cost between the aggregated inference distribution and the prior distribution, in which we don't directly optimize any parameters in the flow-like short-run MCMC model but update or correct the errors in the output. With the corrected inference output, we can update the parameters of the top-down generative model more accurately. As a matter of fact, the trustingly updated model would improve the accuracy of the posterior distribution in return, thus further influencing the short-run MCMC inference in the next step.

Specifically, the algorithm proposed in this paper iterates the following three steps: (i) inference step: inferring the latent variables for each observed example by a short-run Langevin dynamics that aims at drawing samples from the posterior distribution; (ii) correction step: moving the whole population of all the inferred latent vectors to the prior distribution that is assumed in the top-down generative model through optimal transport; (iii) learning step: update the model parameters by gradient descent based on the corrected versions of the inferred latent vectors and the corresponding observed examples.

There are several advantages in the proposed algorithm: (i) efficiency: The learning and inference processes of the model are efficient with a short-run MCMC using a finite number of Langevin steps. Compared to the traditional long-run MCMC inference, the short-run MCMC inference is less time-consuming. In contrast to the variational inference, the short-run MCMC inference is less memory-consuming. (ii) convenience: once the network architecture of the top-down generative model is designed, the approximate inference model represented by the short-run MCMC is automatic and immediately ready in the sense that there is nothing to worry about the design and training of a separate inference model. Both bottom-up inference and top-down generation are governed by the same set of parameters. The unified framework is not only naturally elegant but also statistically rigours. (iii) accuracy: the optimal transport corrects the errors of the non-convergent short-run MCMC inference, thus improves the accuracy of the model parameter estimation.

The contributions of the paper are three-fold: (i) We propose to train a top-down deep generative model or a deep latent variable model by a non-convergent short-run MCMC inference with OT correction. This is the first paper to combine the non-

convergent short-run MCMC and the OT theory to train deep generative models. (ii) We extend the semi-discrete OT algorithm to approximate the one-to-one map between the inferred latent vectors and the samples drawn from the prior distribution in our settings. (iii) We provide strong empirical results in our experiments to verify the effectiveness of the proposed strategy to train deep top-down generative models, including image reconstruction, image generation, anomaly detection, and supervised image inpainting. (iii) Based on our proposed MCMC-OT learning strategy, we further propose an unsupervised learning algorithm to train top-down generative models from incomplete data, such that our algorithm can be useful for unsupervised image inpainting.

2 Related work

In this section, we review prior works related to the proposed framework in our paper.

Variational inference. To avoid the computationally expensive MCMC inference step in the MLE training, variational auto-encoder (VAE) [9] is a popular method to learn top-down generator network by simultaneously training a tractable inference network to approximate the intractable posterior distribution of the latent variables. This is also called reparameterization trick. The VAE, as one of the powerful likelihood-based generative models, has been successfully applied to image generation [36], [37], video generation [38], point cloud generation [39], image captioning [40], saliency prediction [28], continual learning [41], [42], etc. However, in VAE, one needs to design an inference model for the latent variables, which is a non-trivial task in a generator network with complex architecture. Our method does not rely on an extra inference model to assist the training. It performs inference by short-run Langevin sampling from the posterior distribution, followed by an optimal transport correction. Despite the great success of VAEs, several studies have shown that VAE prior fails to match the aggregate approximate posterior [43], [44]. Such a bias due to the approximate inference would lead to undesired regions in the latent space that are not decoded to meaningful data examples. These regions often have densities under the prior distribution but have low densities under the aggregate approximate posterior. The mismatching between the prior and the aggregate approximate posterior causes bad quality of the synthesized data. Although our method doesn't belong to variational inference or VAE, we also study the mismatching problem between the prior and the aggregate posterior of the short-run MCMC inference. In our framework, we propose to use the optimal transport theory to correct mismatching.

Alternating back-propagation algorithm. The maximum likelihood learning of the top-down generator network, including its dynamic version, can be achieved by the alternating back-propagation (ABP) algorithm [11], [30], without resorting to an inference model. The ABP algorithm trains the generator model by alternating the following two steps: (i) inference step: inferring the latent variables for each training example by Langevin sampling from the posterior distribution, and (ii) learning step: updating the model parameters based on the training data and the corresponding inferred latent variables by the gradient descent optimizer. The former step involves the computation of the gradient of the generator network with respect to the latent variables, while the latter step involves the computation of the gradient of the generator network with respect to the parameters. Both steps compute the gradients conveniently and efficiently with the power of back-propagation due to the differentiability of the network. This is the origin of the name of the algorithm. The ABP algorithm has been successfully applied to self-supervised saliency detection [27], zero-shot learning [26], video generation [45], multi-view image generation [46], unsupervised disentanglement of appearance and deformation in images [47], unsupervised disentanglement of appearance, trackable and intrackable motions in videos [31], etc. The ABP algorithm has been extended to model flexible and informative latent prior in [24], where the top-down generator adopts a trainable energy-based model (EBM) [1] as the prior distribution instead of a Gaussian distribution. The usage of the latent EBM prior in the generator leads to a change of the ABP algorithm, which is an extra MCMC sampling step from the EBM prior distribution for estimating the parameters of the EBM. [48] proposes to use a short-run MCMC for approximate inference of latent variables in the inference step of the ABP algorithm, and provides a variational optimization method to determine the optimal step size of the short-run MCMC. Our method also uses a short-run MCMC for inference but adopts a different strategy, i.e., optimal transport, to correct the bias due to the non-convergence of the short-run MCMC inference. Our framework belongs to the ABP family in the sense that it improves the ABP algorithm by using the optimal transport theory.

Optimal Transport (OT). OT is used to compute the distance between two measures and is able to push forward the source distribution to the target distribution [35], [49]. Recently, OT has been widely used in generative models to help generate high quality samples. For example, by replacing the original KL-divergence in the generative adversarial networks (GANs) [6] with the Wasserstein-1 distance, Arjovsky et al. [8] propose the WGAN model to achieve better convergence and generate higher quality samples. To satisfy the Lipschitz condition required by computing the $W1$ distance in the discriminators, Gulrajani et al. [50] use the weight clipping and Miyato et al. [51] propose the spectral normalization. Tolstikhin et al. [52] propose the Wasserstein variational auto-encoder that minimizes the Wasserstein distance between the inference model and the posterior distribution. With the Wasserstein-2 distance, Korotin et al. [53] introduce a novel end-to-end non-minimax algorithm for training the generative models by using the recently proposed Input Convex Neural Networks (ICNNs) [54]. Besides the Wasserstein distance, the optimal transport is also used to transport a simple uniform distribution to the complex latent feature distribution extracted by the autoencoder for image generation [55], [56].

There are typically three settings for the OT problems based on the different source and target distributions. (i) When both the source and target measures are continuous and admit the corresponding density functions, according to the Brenier theorem [57], solving the OT problem is equivalent to finding the solution of the famous Monge-Amperè equation with the squared Euclidean distance [35], [58]. By linearizing this complex variant coefficient elliptic partial differential equation (PDE) in each iteration, it is converted to a positive definite linear system using the finite-difference scheme and can be solved by the BiCG algorithm [59]. Later, Benamou et al. [60] propose to solve this PDE on more general domains using Newton's method. But these algorithms are limited to the low-dimensional OT problems. To solve the high dimensional problem, we can use the recent proposed Input Convex Neural Networks (ICNNs) [54] to approximate the convex Brenier potential. (ii) If the source measure is continuous, and the target measure is discrete, it is called the semi-discrete OT problem. With the squared Euclidean distance, the Brenier potential, whose

gradient gives the OT map, can be represented by a piecewise linear convex function. For low dimensional problems, Gu et al. [61] propose to solve this problem by optimizing a convex energy by computing its gradient and Hessian matrix through convex geometry. Later, An et al. [55] extend this method to solve the high dimensional problems by estimating the gradient through Monte Carlo sampling. (iii) When both the source and target measures are discrete, the discrete OT problem can be directly solved by linear programming with computational complexity $O(n^3)$, which is prohibitively high if n is large. To solve this problem, Cuturi [62] adds an entropy regularizer to the prime problem and then uses the Sinkhorn algorithm to solve the regularized problem with computational complexity $\tilde{O}(n^2/\epsilon^2)$ [63]. Recent methods on solving the discrete OT problems generalize this idea by introducing different regularizers [64], [65] or solving larger scale problems via stochastic method [66], [67]. But unlike the continuous or the semi-discrete settings whose solvers obtain the exact solutions of the OT map, this kind of methods can only give approximate solutions of the OT plan and cannot reconstruct the corresponding OT map.

In the following, we directly solve the Kantorovich dual problem in the discrete settings through stochastic gradient descent, and then recover the approximate OT map efficiently.

3 Maximum Likelihood Learning of Top-Down Generative Model

3.1 Factor analysis model

We start from the factor analysis model, which is a prototype of the top-down generative model. Let $\mathbf{I}$ be a D-dimensional observed data example, such as an image. Let z be the d-dimensional vector of continuous unobserved latent variables, from which the observed data are assumed to be generated. The traditional factor analysis model assumes that each observed data example is a liner transformation of an unobserved latent vector of variables and is modeled by $\mathbf{I} = \Theta z + \epsilon$, where Θ is a $D \times d$ matrix, ϵ is a D-dimensional residual vector following Gaussian distribution, i.e., $\epsilon \sim \mathcal{N}(0, \sigma^2 I_D)$, and the latent vector z also follows Gaussian distribution, i.e., $z \sim \mathcal{N}(0, \sigma^2 I_d)$. I_d and I_D are d-dimensional and D-dimensional identity matrices, respectively. In general, d is assumed to be much less than D, i.e., $d \ll D$. The matrix Θ contains all the parameters of the model, and z is said to be a vector representation (or a code) of $\mathbf{I}$. Suppose we observe a set of data examples $\{\mathbf{I}_i\}$, the goal of learning a factor analysis model is to estimate Θ while inferring $\{z_i\}$, which is an unsupervised learning problem and can be accomplished by maximum likelihood via the expectation-maximization (EM) algorithm.

3.2 Generator Network

Generalizing from traditional factor analysis model, the generator network assumes the observed example $\mathbf{I}$ is generated from a latent vector z by a non-linear transformation $\mathbf{I} = g_\theta(z) + \epsilon$, where g_θ is a top-down convolutional neural network (sometime called deconvolutional neural network) with parameters θ that consist of all trainable weights and bias terms in the network, $\epsilon \sim \mathcal{N}(0, \sigma^2 I_D)$ is the observation error, and $z \sim \mathcal{N}(0, I_d)$. We assume $d \ll D$. The generator network is essentially a non-linear factor analysis model that defines the joint distribution of $(\mathbf{I}, z)$,

$$p_\theta(\mathbf{I}, z) = p_\theta(\mathbf{I}|z)p(z), \tag{1}$$

where we assume the prior distribution $p(z) = \mathcal{N}(0, I_d)$ and $p(\mathbf{I}|z) = \mathcal{N}(g_\theta(z), \sigma^2 I_D)$. The standard deviation σ takes an assumed value. Following the Bayes rule, we can easily obtain the marginal distribution $p_\theta(\mathbf{I}) = \int p_\theta(\mathbf{I}, z)dz$, and the posterior distribution $p_\theta(z|\mathbf{I}) = p_\theta(\mathbf{I}, z)/p_\theta(\mathbf{I}) = p_\theta(\mathbf{I}, z)/\int p_\theta(\mathbf{I}, z)dz$.

3.3 Maximum likelihood learning

Given a set of training examples $\{\mathbf{I}_i, i = 1, \ldots, n\} \sim p_{\text{data}}(\mathbf{I})$, where $p_{\text{data}}(\mathbf{I})$ is the unknown data distribution. We can train p_θ by maximizing the log-likelihood of the training samples

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log p_\theta(\mathbf{I}_i), \tag{2}$$

which is equivalent to minimizing the KL-divergence between the true data distribution $p_{\text{data}}(\mathbf{I})$ and the model $p_\theta(\mathbf{I})$ when the number of training examples n is large enough [11]. To be specific,

$$\begin{aligned} &\text{KL}(p_{\text{data}}(\mathbf{I})||p_\theta(\mathbf{I})) \\ =&\text{E}_{p_{\text{data}}(\mathbf{I})}\left[\log \frac{p_{\text{data}}(\mathbf{I})}{p_\theta(\mathbf{I})}\right] \\ =&\text{E}_{p_{\text{data}}(\mathbf{I})}[\log p_{\text{data}}(\mathbf{I})] - \text{E}_{p_{\text{data}}(\mathbf{I})}[\log p_\theta(\mathbf{I})], \end{aligned} \tag{3}$$

where the left term is the entropy of the data distribution that is independent to the model parameter θ, therefore we have

$$\begin{aligned} &\arg\min_\theta \text{KL}(p_{\text{data}}(\mathbf{I})\|p_\theta(\mathbf{I})) \\ =&\arg\max_\theta \text{E}_{p_{\text{data}}(\mathbf{I})}[\log p_\theta(\mathbf{I})] \\ \approx&\arg\max_\theta \frac{1}{n}\sum_{i=1}^{n} \log p_\theta(\mathbf{I}_i) = \arg\max_\theta \mathcal{L}(\theta), \end{aligned} \tag{4}$$

where we use the law of large number to obtain the final equation. Eq. (4) provides an interpretation of the behavior of MLE, i.e., maximizing the data likelihood of the model is equal to minimizing the difference between the model and the data distribution. We can see that MLE is a proxy to fit the model to the data distribution, which cannot be achieved directly because the data distribution is unknown to us.

The maximization of the log-likelihood function presented in Eq. (2) can be accomplished by gradient ascent algorithm that iterates

$$\theta_{t+1} = \theta_t + \gamma_t \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta \log p_\theta(\mathbf{I}_i), \tag{5}$$

where γ_t is the learning rate depending on time t and is typically scheduled to decay over time. The gradient of the log probability is given by

$$\begin{aligned} \nabla_\theta \log p_\theta(\mathbf{I}) &= \frac{1}{p_\theta(\mathbf{I})}\nabla_\theta p_\theta(\mathbf{I}) \\ &= \frac{1}{p_\theta(\mathbf{I})}\nabla_\theta \left[\int p_\theta(\mathbf{I}, z)dz\right] \\ &= \int [\nabla_\theta \log p_\theta(\mathbf{I}, z)] \frac{p_\theta(\mathbf{I}, z)}{p_\theta(\mathbf{I})}dz \\ &= \text{E}_{p_\theta(z|\mathbf{I})}[\nabla_\theta \log p_\theta(\mathbf{I}, z)]. \end{aligned} \tag{6}$$

$\text{E}_{p_\theta(z|\mathbf{I})}[\cdot]$ denotes the expectation under the posterior distribution $p_\theta(z|\mathbf{I})$, which is analytically intractable and can be approximated by MCMC. To compute $\nabla_\theta \log p_\theta(\mathbf{I})$ in Eq. (6), we need to estimate the gradient of the logarithm of the joint distribution with

respect to the model parameters $\nabla_\theta \log p_\theta(\mathbf{I}, z)$. According to Eq. (1), the logarithm of the joint distribution is given by

$$\log p_\theta(\mathbf{I}, z) = -\frac{1}{2\sigma^2}\|\mathbf{I} - g_\theta(z)\|^2 - \frac{1}{2}\|z\|^2 + \text{const}, \quad (7)$$

where the constant term is independent of the latent vector z or parameters θ, thus $\nabla_\theta \log p_\theta(\mathbf{I}, z) = \frac{1}{\sigma^2}(\mathbf{I} - g_\theta(z))\nabla_\theta g_\theta(z)$, where $\nabla_\theta g_\theta(z)$ can be efficiently computed by back-propagation.

4 Short-run MCMC inference

4.1 Long-run Langevin dynamics

To learn the model parameter θ by using Eq. (5), the key is to compute the intractable expectation term in Eq. (6), which can be achieved by first drawing samples from $p_\theta(z|\mathbf{I})$ and then using the Monte Carlo sample average to approximate it. Given a step size $s > 0$, and an initial value z^0, Langevin dynamics [33], [68], which is a gradient-based MCMC method, can produce samples from the posterior density $p_\theta(z|\mathbf{I})$ by recursively computing

$$z^{k+1} = z^k + \frac{s^2}{2}\nabla_z \log p_\theta(z|\mathbf{I}) + s\xi_k, \quad (8)$$

where k indexes the time step of Langevin dynamics, $\xi_k \sim \mathcal{N}(0, I_d)$ is a random noise diffusion that helps escape from local modes. Also, $\nabla_z \log p_\theta(z|\mathbf{I}) = \frac{1}{\sigma^2}(\mathbf{I} - g_\theta(z))\nabla_z g_\theta(z) - z$, where $\nabla_z g_\theta(z)$ can be efficiently computed by back-propagation.

Let us use K to denote the number of Langevin steps. When $s \to 0$ and $K \to \infty$, no matter what the initial distribution of z^0 is, z^K will converge to the posterior distribution $p_\theta(z|\mathbf{I})$ and become a fair sample from $p_\theta(z|\mathbf{I})$. A Metropolis-Hastings step may be added to correct for the finite step size s, but this step is often ignored in practice, such as [11], [26], [27], [30], for the purpose of efficient computation.

4.2 Short-run Langevin dynamics

With limited affordable computational resources, it is not sensible or realistic to use a long-run MCMC to train the model. Also, the target posterior distribution that we sample can be highly complex such that the Langevin chains have no hope to converge. Therefore, within each iteration, running a finite number of Langevin steps for inference toward $p_\theta(z|\mathbf{I})$ appears to be practical and inevitable. Thus, a short-run K-step Langevin dynamics is given by

$$\begin{aligned} &z^0 \sim p_0(z), \\ &z^{k+1} = z^k + \frac{s^2}{2}\nabla_z \log p_\theta(z|\mathbf{I}) + s\xi_k, k = 1, .., K. \end{aligned} \quad (9)$$

The initial distribution $p_0(z)$ is assumed to be the Gaussian white noise distribution in this paper. Following [12], such a dynamics can be treated as a conditional generator that transforms a random noise z^0 to the target distribution under the condition $\mathbf{I}$. And the transformation itself can also be treated as a K-layer residual network, where each layer shares the same parameters θ and has a noise injection. We use κ_θ to denote the K-step MCMC transition kernel. The conditional distribution of z^K given $\mathbf{I}$ is

$$q_\theta(z^K|\mathbf{I}) = \int p_0(z^0)\kappa_\theta(z^K|z^0, \mathbf{I})dz^0, \quad (10)$$

and the corresponding marginal distribution of z^K, or also called the aggregated posterior distribution, is

$$q_\theta(z^K) = \int q_\theta(z^K|\mathbf{I})p_{\text{data}}(\mathbf{I})d\mathbf{I}. \quad (11)$$

If the MCMC converges, $q_\theta(z^K)$ should be close to the prior distribution $p(z)$, otherwise there is a gap between them. A short-run MCMC with finite steps of Langevin update is certainly a non-convergent MCMC since each z^K is highly dependent on its initialization z_0. Training a top-down generative model with a non-convergent MCMC inference will cause a biased estimation of the model parameters. Especially, using non-convergent $\{z_i^K\}$ as inferred latent vectors to update θ in the learning stage will lead to a failure of data generation initialized from the prior in the testing stage. The reason is because the generator network g_θ is trained to connect the samples from the biased aggregated posterior distribution $q_\theta(z^K)$, which deviates from the prior, and the data examples from the data distribution $p_{\text{data}}(\mathbf{I})$. There is no way to use such a biased generator network to synthesize realistic examples by transforming random samples from the prior distribution $p_0(z)$.

Eq. (9) is also called the noise-initialized short-run MCMC, where for each step of parameter update, the short-run MCMC starts from the noise distribution $z^0 \sim p_0(z)$. If the short-run MCMC is initialized by the inferred results obtained in previous iteration, it is called the persistent short-run MCMC.

Despite the efficiency of the short-run MCMC inference in Eq. (10), it might not converge to the true posterior distribution $p_\theta(z|\mathbf{I})$. Some prior works have started to investigate how to address the discrepancy between prior and aggregated posterior. For example, [12] treats the short-run MCMC as an approximate inference model and optimizes the step size s by variational inference, in which the step size s is optimized via either a grid search or gradient descent, so that the short-run MCMC $q_s(z|\mathbf{I})$ (here s is the learning parameter) can best approximate the posterior distribution $p_\theta(z|\mathbf{I})$. Our paper focuses on the same goal to deal with the bias of the short-run MCMC inference in the context of learning top-down generative models.

5 MCMC inference with OT correction

In this paper, we propose to use optimal transport to correct the bias of the short-run inference results. Instead of minimizing the difference between the short-run inference model and the true posterior, i.e., $\text{KL}(q_\theta(z^K|\mathbf{I})|p_\theta(z|\mathbf{I}))$, we use OT to minimize the transport cost between the aggregated posterior distribution $q_\theta(z^K)$ of the latent variables inferred by the short-run Langevin dynamics and the prior distribution $p_0(z)$.

5.1 OT correction for biased short-run MCMC

To be specific, for learning a top-down generative model $\mathbf{I} = g_\theta(z)$ that generates an observed image $\mathbf{I}$ from a latent vector z, we iterate the following three steps.

1) **Inference step**: we first infer the latent vector for each observed image $\mathbf{I}_i$ by a K-step short-run MCMC, i.e., $\hat{z} \sim q_\theta(z^K|\mathbf{I}_i)$, and then we obtain a population $\{\hat{z}_i\}$ of the inferred latent vectors for all observed data $\{\mathbf{I}_i\}$, where $\{\hat{z}_i\} \sim q_\theta(z^K)$;
2) **Correction step**: We use OT to move $\{\hat{z}_i\}$ to the desired prior distribution for closing the gap between them due to non-convergent inference. The OT reshapes the biased population to the prior distribution with a minimum moving cost. With the more correct inferred latent vectors, the subsequent parameter update can be more accurate;
3) **Learning step**: Given the observed images and their corresponding inferred latent vectors, we update θ by following

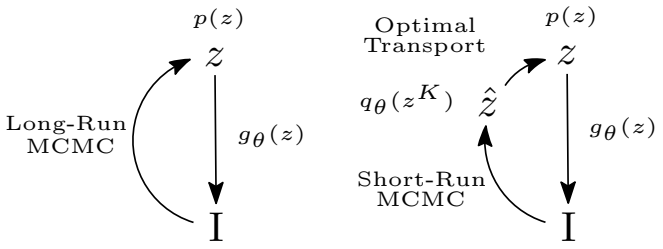


Fig. 1. Diagrams of two learning strategies for latent variable models: (left) the long-run MCMC inference framework. (right) the proposed framework using a short-run MCMC with OT correction.

Eq. (5) and Eq. (6). As the θ becomes increasingly well-trained, the inference engine $q_\theta(z^K)$ becomes more accurate and the correction made by OT also becomes smaller.

An illustration of the proposed strategy is presented in Fig. 1, where we also compare our framework with the one using a traditional long-run MCMC inference.

In practise, we can use either the noise-initialized short-run MCMC or the persistent short-run MCMC in the inference step. In our experiment we choose the latter one for the purpose of quick convergence. As to the correction stage, we learn the one-to-one OT map from $\{\hat{z}_i\}$ to $\{z_i\}$, which is a population sampled from the prior Gaussian distribution and of the same size as $\{\hat{z}_i\}$. Computing the optimal transport at each iteration is time-consuming and unnecessary in practise. To make the whole pipeline more efficient, we actually perform the correction step after every L iterations. After we get the bijective OT map $T(\hat{z}_i) = z_j$, instead of directly updating the model through the paired data $\{(T(\hat{z}_i), \mathbf{I}_i)\}$, we choose to correct $\hat{z}_i$ by using a mixture of the OT result and the old one to avoid unstable learning due to a sudden change of $\hat{z}_i$, i.e.,

$$\hat{z}_i \leftarrow \alpha T(\hat{z}_i) + (1-\alpha)\hat{z}_i, \tag{12}$$

where $\alpha \in [0, 1]$ is a hyperparameter that controls the percentage of the OT result used for correction. Then we get the corrected paired data $\{(\hat{z}_i, \mathbf{I}_i)\}$, which are used to update the model parameter θ. Note that when $\alpha = 0$, our model degenerates to the traditional ABP model [11]. If α is set to be 1, we correct the short-run outputs totally with the OT results. A moderate $0 < \alpha < 1$ is typically helpful to gradually pull the marginal distribution $q_\theta(z^K)$ to the prior distribution $p(z)$ for ensuring a smooth correction. We summarize the whole pipeline of our learning strategy in Alg. 1.

Algorithm 1 Short-run MCMC inference with OT correction

Input:
(1) observed examples $\{\mathbf{I}_i\}$,
(2) number of skip steps L,
(3) number of Langevin steps K,
(4) Langevin step size s,
(5) random samples $\{z_j\}$ from the prior distribution $\mathcal{N}(0, I_d)$,
(6) hyperparameter α.
Output: Model parameters θ.
$k \leftarrow 1$
repeat
  # Inference
  Infer the latent vectors $\{\hat{z}_i\}$ from $\{\mathbf{I}_i\}$ by a K-step short-run Langevin dynamics in Eq. (9). The short-run MCMC can be initialized by random noise or the previous result.
  # Correction
  if $k\%L == 0$ **then**
    Compute the approximate OT map $\hat{T}$ from $\{\hat{z}_i\}$ to $\{z_j\}$ according to Alg. 2.
    $\hat{z}_i \leftarrow \alpha\hat{T}(\hat{z}_i) + (1-\alpha)\hat{z}_i$
  end if
  # Learning
  Update the model parameter θ by following Eq. (5) and Eq. (6) with the paired data $\{(\hat{z}_i, \mathbf{I}_i)\}$.
  $k \leftarrow k + 1$
until Converge

Algorithm 2 Optimal Transport

Input: source samples $\{\hat{z}_i\}_{i=1}^n$, target samples $\{z_j\}_{j=1}^n$, and a threshold ϵ.
Output: $\hat{T}$
Initialize $h = (0, 0, \ldots, 0)$.
repeat
  Compute J_j for $j = 1, 2, \ldots, n$
  Compute $\frac{\partial E}{\partial h_j} = \frac{\#J_j}{n} - \frac{1}{n}$
  Update h according to the Adam algorithm with $\beta_1 = 0.9$ and $\beta_2 = 0.5$.
until $\|\nabla E\| \leq \epsilon$
Build the approximate OT map $\hat{T}$ through $J_j,\ j = 1, 2, \ldots, n$.

5.2 Optimal transport

Given the latent codes sampled from $q_\theta(z^K)$, namely $\{\hat{z}_i\}_{i=1}^n$, and the randomly generated samples $\{z_j\}_{j=1}^n$ from the prior $\mathcal{N}(0, I_d)$, the one-to-one map from $\{\hat{z}_i\}$ to $\{z_j\}$ is computed through the optimal transport. Specifically, we set the cost function to be the squared Euclidean distance $c_{ij} = \|\hat{z}_i - z_j\|_2^2$ because it has a beautiful geometric meaning [58], and then solve the following assignment problem:

$$\min_{\pi \in \Pi} \sum_{i,j=1}^{n} \pi_{ij} c_{ij} \tag{13}$$

where $\Pi = \{\pi | \sum_{j=1}^n \pi_{ij} = \frac{1}{n}, \sum_{i=1}^n \pi_{ij} = \frac{1}{n}, \pi_{ij} \geq 0\}$. According to the linear programming theory, there will be only one nonzero element in each row/column of π. Actually, all of the nonzero elements should be equal to $1/n$. Thus, we can define the map from $\{\hat{z}_i\}$ to $\{z_j\}$ like this: $T : \hat{z}_i \rightarrow z_j$ if $\pi_{ij} \neq 0$. When n is large, directly solving the above problem with Linear Programming will be problematic, since the computational complexity is prohibitively high ($O(n^{2.5})$ according to [69]). Similarly, the classical Hungarian algorithm [70] for the assignment problem cannot be used to solve this problem due to the high computational complexity $O(n^3)$. It is also impossible to solve the above problem with the approximate OT solvers, e.g., the Sinkhorn algorithm [62], since these solvers tend to give a dense transport plan, from which it is impossible to recover the OT map. Moreover, the approximate algorithms are not suitable for large scale problems with $n > 20,000$. Thus, we turn to the dual problem of Eq. (13). Here we extend the original dual formula for the semi-discrete OT in [55], [61], [71] to the following minimization problem in our discrete setting:

$$\min_h E(h) = \frac{1}{n}\sum_{j=1}^{n} \max_j \{\langle \hat{z}_i, z_j \rangle + h_j\} - \frac{1}{n}\sum_{j=1}^{n} h_j. \tag{14}$$

The above problem is convex as it is the maximum of the summation of n hyperplanes. Thus, it can be solved by the gradient descent algorithm. The gradient is computed by $\frac{\partial E}{\partial h_j} = \frac{\#J_j}{n} - \frac{1}{n}$, where $J_j = \{i | \langle \hat{z}_i, z_j \rangle + h_j \geq \langle \hat{z}_i, z_k \rangle + h_k \ \forall \ k \in [n]\}$ and $\#J_j$ is the number of elements in J_j. Assume h^* is an optimal solution of $E(h)$, then $h = h^* + (c, c, \ldots, c)^T$ is also an optimal solution. To omit the ambulation, we define $\nabla E(h) = \nabla E(h) - \text{mean}(\nabla E(h))$. With the gradient information, the energy $E(h)$ can be minimized by the Adam gradient descent algorithm [72].

Since Eq. (14) is the dual of the assignment problem, with the optimal solution h^*, it is easy to reconstruct the one-to-one OT map from $\{\hat{z}_i\}$ to $\{z_j\}$ by $T : \hat{z}_i \rightarrow z_j, j = \arg\max_k \langle \hat{z}_i, z_k \rangle + h_k^* \ \forall \ k \in [n]$. During the optimization process, we stop when the norm of the gradient $\nabla E(h)$ is less than ϵ. Ideally, if $\epsilon = 0$, the map T will be injective and surjective, and each J_j only includes one element, namely the corresponding i. In that case, the OT map T is well defined. In reality, we usually set $\epsilon > 0$, therefore T will be neither injective nor surjective. In such a situation, for some z_js, there may be one or more corresponding $\hat{z}_i$s; and for some other z_js, the corresponding $\hat{z}_i$s may not exist. To omit the ambiguity and reconstruct the one-to-one map, we need to handle the set J_j that will be empty or include one or more elements. The approximate OT map $\hat{T}$ is thus given by: (i) if there is only one element in J_j, namely i, then $\hat{T}(\hat{z}_i) = z_j$; (ii) when J_j includes more than one elements, we randomly select $i \in J_j$ and abandon the others, then define $\hat{T}(\hat{z}_i) = z_j$; (iii) the abandoned $\hat{z}_i$s and the z_js corresponding to the empty J_js are removed from the domain and range of $\hat{T}$, respectively. In such a way, we build a new injective and surjective map $\hat{T}$ that approximates the OT map T well.

Note that in our OT algorithm, the prior distribution is not limited to the Gaussian distribution. We can actually choose any prior distribution as long as it is easy to sample from. Additionally, the computational complexity to solve the nonsmooth dual problem in Eq. (14) is $O(n^2/\sqrt{\epsilon})$ [73]. Under the background of training the complex neural networks with a large number of parameters, the time used to optimize the OT problem is negligible. Finally, the number of the removed samples from $\hat{T}$ should not be larger than $n\epsilon$. In our experiments, we usually set $\epsilon = 0.05$. With such a small ϵ, we can get a good approximation of the OT map.

6 Learning from Incomplete Data

A major advantage of a top-down generative model is to learn from incomplete data, where each data example is partially observed. (For example, some pixels of each training image are occluded or unobserved.) Learning the top-down generative model from incomplete data can be considered a non-linear generalization of matrix completion. In this section, we will show that, by making a small modification, we can generalize the proposed MCMC-OT inference algorithm above to the scenario of training models from incomplete data. Recall that in the scenario of learning from complete data or fully observed data, the learning objective, i.e., maximum likelihood in Eq. (2), is computed by summing over all the pixels of the images, while in the setting of learning from partially visible images, we will instead compute the likelihood by summing over only the visible pixels of the images in the sense that we estimate the model parameters by maximizing the likelihood of the visible pixels.

Formally, suppose we observe a set of incomplete training examples $\{(\mathbf{I}_i, M_i), i = 1, ..., n\}$, where we use M_i to denote the known positions of the missing information in each training example $\mathbf{I}_i$. Specifically, M_i is a matrix, whose number of dimension is the same as that of the image $\mathbf{I}_i$, with values ones indicating the visible pixels and zeros indicating the invisible (missing, corrupted, or unobserved) pixels of the image $\mathbf{I}$, respectively. We learn the model, i.e., parameters θ, from the incomplete data by maximizing

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \log p_\theta(\mathbf{I}_i | M_i). \tag{15}$$

The joint distribution is now given by

$$\log p_\theta(\mathbf{I}, z | M) = -\frac{1}{2\sigma^2} \| M \odot (\mathbf{I} - g_\theta(z)) \|^2 - \frac{1}{2} \|z\|^2 + \text{const}, \tag{16}$$

where $\odot$ is the element-wise multiplication operator. The joint distribution presented in Eq. (16) is the key to compute

$$\nabla_\theta \log p_\theta(\mathbf{I}, z | M) = \frac{1}{\sigma^2} (M \odot (\mathbf{I} - g_\theta(z))) \nabla_\theta g_\theta(z), \tag{17}$$

which is the learning gradient to update the model parameter in the scenario of learning from missing data, and

$$\nabla_z \log p_\theta(z | \mathbf{I}, M) = \frac{1}{\sigma^2} (M \odot (\mathbf{I} - g_\theta(z))) \nabla_z g_\theta(z) - z, \tag{18}$$

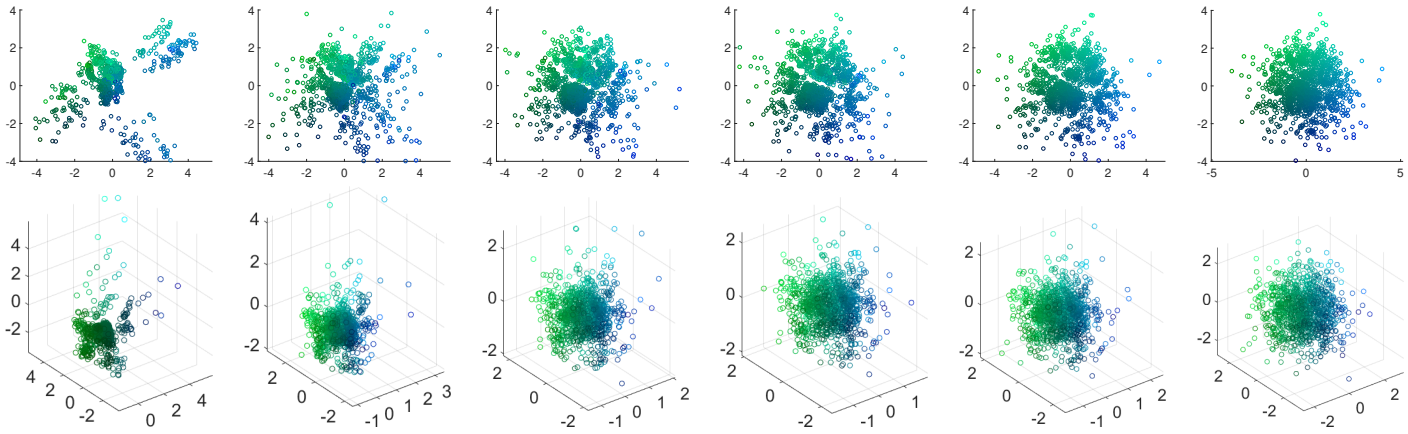
which is derived from Eq. (16) and is the sampling gradient to infer latent variables from the incomplete data by Langevin dynamics

$$\begin{aligned} &z^0 \sim p_0(z), \\ &z^{k+1} = z^k + \frac{s^2}{2} \nabla_z \log p_\theta(z | \mathbf{I}, M) + s\xi_k, k = 1, .., K. \end{aligned} \tag{19}$$

The OT correction step is the same as the one in the original algorithm since there is no missing information in the latent vectors so that Alg.2 is still applicable. Each incomplete training example is completed or recovered by first inferring the latent vector via Eq. (18) and then transforming the inferred latent vector to data. As to image recovery or inpainting in this scenario, we always fix the visible part of the incomplete example and only update the values in the invisible part. Since the model never sees the ground truth intensities of the invisible pixels during training, this is a task of unsupervised image inpainting. We can slightly modify Alg.1 to obtain a full description of the proposed learning algorithm for incomplete data in Alg. 3. The ability to learn from incomplete data can be considered as a criterion to evaluate a generative model.

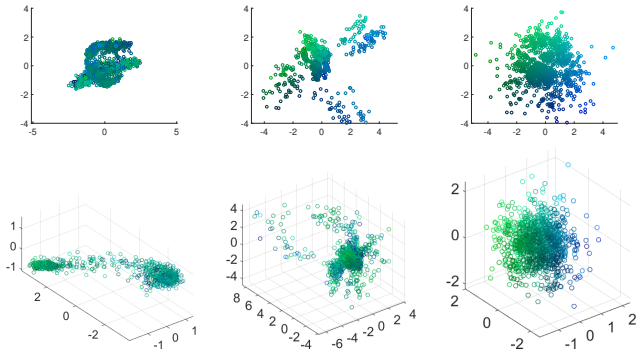
7 Experiments

In the experiments, we test the proposed model in terms of whether it can (i) successfully correct the marginal distribution $q_\theta(z^K)$ of the latent vectors inferred by the short-run Langevin dynamics, (ii) learn an expressive generator that synthesizes visually realistic images from the prior distribution, (iii) perform image inpainting with the learned generator, (iv) successfully perform anomaly detection, and (v) perform unsupervised image recovery by learning from the incomplete images. To show the performance of our method, we experiment on MNIST [74], SVHN [75] and CelebA [76] datasets. Moreover, to investigate the influence of different hyperparameters, we mainly use the MNIST dataset due to its simplicity and representativeness. To quantify the performance of the model, we adopt the mean squared error (MSE) and the FID score [77] to measure the quality of the reconstructed



(a) 1st OT (a) 2nd OT (a) 4th OT (a) 6th OT (a) 8th OT (a) Prior

Fig. 2. Visualization of the marginal distribution of the inferred latent codes $q_\theta(z^K)$ obtained by the short-run MCMC inference at different iterations, as well as the prior distribution. The first row shows the results of the experiments using training images from the classes "0" and "1" of the MNIST dataset with the latent dimension being 2. The second row shows the results of the experiments using training images from the classes "T-shirt" and "Trouser" of the Fashion-MNIST dataset with the latent dimension being 3. We plot samples from $q_\theta(z^K)$ at iterations where OT correction is performed.



(a) VAE [9] (b) ABP [11] (c) Ours

Fig. 3. Comparison of the marginal distributions of z inferred by different models trained on images selected from classes "0" and "1" of MNIST dataset (the first row) and classes "T-shirt" and "Trouser" of the Fashion-MNIST dataset (the second row).

and generated images. The MSE loss is also used to evaluate the performance of the model in the tasks of learning from incomplete data for unsupervised image recovery.

7.1 Experimental Settings

Datasets In the experiments, we mainly use the MNIST datatset [74] ($28 \times 28 \times 1$), SVHN dataset [75] ($32 \times 32 \times 3$) and CelebA dataset [76] ($64 \times 64 \times 3$). For the first two datasets, we use all of the samples in the training set, namely 60,000 for the MNIST dataset and 73,257 for the SVHN dataset. For the CelebA dataset, we randomly select 60,000 images for the purpose of quick convergence. For the task of learning from incomplete data, we randomly pick 10,000 images to conduct the experiments. All of the training images are resized and scaled to the range of $[0, 1]$.

Model architectures The architectures of the models are presented in Tab. 1, where the numbers of latent dimensions are set to be $30, 64, 64$ for the MNIST dataset, SVHN dataset and CelebA dataset, respectively. We use the same architecture for the CelebA dataset, in both tasks of learning generator from complete data and learning from incomplete data for unsupervised image recovery.

Optimization The parameters for the generators are initialized with Xavier normal [78] and then optimized with the Adam optimizer [72] with $\beta_1 = 0.5$ and $\beta_2 = 0.99$. For all of the experiments, we set the batch size to be 2,000. In Alg. 1 of the paper, both L and K are set to be 50. The hyperparameter α is set to be 0.5 for the MNIST dataset, and 0.3 for the SVHN and CelebA datasets. The step sizes s for MNIST, SVHN and CelebA datasets are set to be 0.3, 3.0, 3.0, respectively. We also set $\sigma = 0.3$ for all of the models.

Algorithm 3 Short-run MCMC inference with OT correction for learning from incomplete data

Input: (1) observed incomplete images and positions of missing pixels $\{\mathbf{I}_i, M_i\}$, (2) number of skip steps L, (3) number of Langevin steps K, (4) Langevin step size s, (5) random samples $\{z_j\}$ from the prior distribution $\mathcal{N}(0, I_d)$, and (6) hyperparameter α.
Output: Model parameters θ.
$k \leftarrow 1$
repeat
  # Inference from incomplete data
  Infer the latent vectors $\{\hat{z}_i\}$ from $\{\mathbf{I}_i, M_i\}$ by a K-step short-run Langevin dynamics in Eq. (19). The short-run MCMC can be initialized by random noise or the previous result.
  # Correction via optimal transport
  if $k\%L == 0$ **then**
    Compute the approximate OT map $\hat{T}$ from $\{\hat{z}_i\}$ to $\{z_j\}$ according to Alg. 2.
    $\hat{z}_i \leftarrow \alpha\hat{T}(\hat{z}_i) + (1 - \alpha)\hat{z}_i$
  end if
  # Learning from incomplete data
  Update the model parameter θ with the gradient computed by Eq. (17) with the triplet $\{(\hat{z}_i, \mathbf{I}_i, M_i)\}$.
  $k \leftarrow k + 1$
until Converge

TABLE 1
The architectures of the generators for different datasets.

Model	Layer	Output	Stride	Padding	BN
MNIST	Input z	30	-	-	-
	Linear, ReLU	1024	-	-	Yes
	Linear, ReLU	7×7×128	-	-	Yes
	2×2 convT, ReLU	14×14×64	2	-	Yes
	2×2 convT, Tanh	28×28×1	2	-	No
SVHN	Input z	64	-	-	-
	Linear, ReLU	2×2×512	-	-	Yes
	5×5 convT, ReLU	4×4×256	2	2	Yes
	5×5 convT, ReLU	8×8×128	2	2	Yes
	5×5 convT, ReLU	16×16×64	2	2	Yes
	5×5 convT, Tanh	32×32×3	2	2	No
CelebA	Input z	64	-	-	-
	Linear, ReLU	4×4×1024	-	-	Yes
	5×5 convT, ReLU	8×8×512	2	2	Yes
	5×5 convT, ReLU	16×16×256	2	2	Yes
	5×5 convT, ReLU	32×32×128	2	2	Yes
	5×5 convT, Tanh	64×64×3	2	2	No

7.2 Latent space analysis

We first conduct experiments to verify that the proposed method does correct the bias of the marginal distribution of the latent variables $q_\theta(z^K)$ obtained by the short-run MCMC inference. For simplicity, we select two classes of images, i.e.,"0" and "1", from the MNIST dataset, and train our model on these images without using label information. We set the number of dimension of the latent space to be 2 for better visualization. We first show the evolution of $q_\theta(z^K)$ by plotting the samples of $q_\theta(z^K)$ at different learning iterations of our model in the first row of Fig. 2. Since we perform OT correction every $L = 50$ learning iterations, we plot samples of $q_\theta(z^K)$ at some selected iterations where OT correction is performed in Fig. 2. We can see that $q_\theta(z^K)$ gradually moves toward the prior distribution due to the OT correction, and finally matches it. The first row of Fig. 3 shows a comparison of the latent vectors inferred by the VAE model [9], the ABP model [11] and our model, respectively. The distributions of latent vectors inferred by the VAE and the ABP models are far from the prior distribution (i.e., a Gaussian distribution), while the marginal distribution of the inferred latent variables $q_\theta(z^K)$ of our model looks much closer to the prior distribution. We conduct one more experiment on images of the classes "T-shirt" and "Trouser" of the Fashion-MNIST dataset. We learn our model with a 3-dimensional latent space. The second row of Fig. 2 displays the evolution of $q_\theta(z^K)$ using our method. The second row of Fig.3 shows a comparison of the VAE, the ABP and our model. We have the same finding as the one we get in the MNIST dataset using a 2-dimensional latent space.

7.3 Image generation and reconstruction

A well-trained top-down generative models can perform data generation via ancestral sampling and data reconstruction via inference of latent variables. A correct or unbiased inference step is crucial in learning a top-down generative model that can synthesize realistic data. More specifically, the update of the generator network highly relies on the inferred latent variables. Therefore, if the marginal distribution of the inferred latent variables $q_\theta(z^K)$ matches the prior distribution very well, then the generator network can be trained as a probability transformation from the prior Gaussian distribution to the data distribution. In this way, we can easily synthesize a high quality image by $\mathbf{I} = g_\theta(z)$ with a latent vector z sampled from the prior Gaussian distribution. Updating the generator with biased inferred latent vectors will result in a disconnection between prior distribution and data distribution.

We test our method on the tasks of image synthesis and image reconstruction, and evaluate the performance in terms of the quality of both the generated and reconstructed images. In the following, we compare our model with some likelihood-based top-down generative models, including (i) variational inference models, such as VAE [9] and its variants 2sVAE [79], RAE [80] and Ladder-VAE [81], (ii) flow-based models, such as Real NVP [82] and GLOW [83], and (iii) other top-down generative models using MCMC-based inference, including the ABP model [11], SRI model [12], whose generator has multiple layers of latent variables, and LEBM model [24], which uses an energy-based model [1], instead of a simple Gaussian distribution, to be an informative prior distribution.

In Fig. 4, we show both the reconstructed images and the generated images with the latent vectors sampled from the given prior distribution. It is obvious that the generated images shown in the second column are realistic and comparable to the real ones in the training datasets. In Table 2, we use the mean squared

reconstruction synthesis

Fig. 4. The reconstructed images (the first column) and the generated images (the second column) of datasets MNIST [74] with a resolution of 28×28 pixels (the first row), SVHN [75] with a resolution of 32×32 pixels (the second row), and CelebA [76] with a resolution of 64×64 pixels (the third row).

TABLE 2
The comparison results on different datasets. The MSE and FID (smaller is better) are used to test the quality of the reconstructed and generated images, respectively.

Models		Variational Inference				Normalizing Flow		Short Run MCMC				
		VAE [9]	2sVAE [79]	RAE [80]	Ladder-VAE [81]	RealNVP [82]	GLOW [83]	ABP [11]	SRI [12]	SRI (L=5) [12]	LEBM [24]	Ours
MNIST	MSE	0.023	0.026	0.015	0.018	-	-	-	0.019	0.015	-	**0.0008**
	FID	19.21	18.81	23.92	-	-	66.04	39.12	-	-	-	**14.28**
SVHN	MSE	0.019	0.019	0.014	0.014	-	-	-	0.018	0.011	0.008	**0.002**
	FID	46.78	42.81	40.02	39.26	103.8	65.27	49.71	44.86	35.23	29.44	**19.48**
CelebA	MSE	0.021	0.021	0.018	0.028	-	-	-	0.020	0.015	0.013	**0.010**
	FID	65.75	49.70	40.95	53.40	58.6	39.84	51.50	61.03	47.95	37.87	**29.75**

TABLE 3
AUPRC scores (larger is better) for unsupervised anomaly detection on the MNIST dataset. Numbers are taken from [24] and results for our model are averaged over 10 experiments for variance.

Heldout Digit	1	4	5	7	9
VAE [9]	0.063	0.337	0.325	0.148	0.104
MEG [84]	0.281 ± 0.035	0.401 ± 0.061	0.402 ± 0.062	0.290 ± 0.040	0.342 ± 0.034
Bigan-σ [85]	0.287 ± 0.023	0.443 ± 0.029	0.514 ± 0.029	0.347 ± 0.017	0.307 ± 0.028
EnGAN [86]	0.281 ± 0.035	0.401 ± 0.061	0.402 ± 0.062	0.29 ± 0.040	0.342 ± 0.034
EBM-VAE [87]	0.297 ± 0.033	0.723 ± 0.042	0.676 ± 0.041	0.490 ± 0.041	0.383 ± 0.025
LEBM [24]	0.336 ± 0.008	0.630 ± 0.017	0.619 ± 0.013	0.463 ± 0.009	0.413 ± 0.010
ABP [11]	0.095 ± 0.028	0.138 ± 0.037	0.147 ± 0.026	0.138 ± 0.021	0.102 ± 0.033
Ours ($\alpha = 0.1$)	0.321 ± 0.024	0.621 ± 0.028	0.686 ± 0.024	$\mathbf{0.622 \pm 0.059}$	0.524 ± 0.041
Ours ($\alpha = 0.3$)	$\mathbf{0.353 \pm 0.021}$	$\mathbf{0.770 \pm 0.024}$	$\mathbf{0.726 \pm 0.030}$	0.550 ± 0.013	$\mathbf{0.555 \pm 0.023}$
Ours ($\alpha = 0.5$)	0.297 ± 0.012	0.695 ± 0.036	0.665 ± 0.029	0.580 ± 0.037	0.497 ± 0.025

error (MSE) to measure the quality of the reconstruction and the Fréchet inception distance (FID) [77] to quantify the quality the generated images. From the table we can find that the proposed method outperforms the other baseline methods in the tasks of reconstruction and generation, which verifies the effect of the OT correction in learning generative models with short-run MCMC inference.

We also display synthesized images generated by the models SRI [12] and LEBM [24] in Fig.5 for qualitative comparison. The SRI and LEBM are closely related to our model because both of them are based on MLE with short-run MCMC inference. We can see that our method generates much sharper and clearer images than they do.

Due to the involvement of the short-run MCMC and the optimal transport, it is necessary to consider the running time of the whole pipeline. Here we take the SVHN dataset which includes 73,257 images with the size $32 \times 32 \times 3$ as an example. We train our model with two NVIDIA TitanX GPUs. For each iteration in Alg. 1, the inference step with $K = 30$ Langevin steps takes about 124 minutes, the correction step by optimal transport takes about 10 minutes and the learning step takes 5 minutes. Generally, we need to run $10 \sim 15$ epochs for the model, which will consume about one day.

Fig. 5. The generated images of methods SRI [12] (the first column) and LEBM [24] (the second column). The first row shows the synthesized images generated by the models that learn from the SVHN dataset [75] with a resolution of 32×32 pixels, and the second row shows the results generated by the models that learn from the CelebA dataset [76] with a resolution of 64×64 pixels.

7.4 Image inpainting

Once the latent variable model is trained from the fully observed images, it can be applied to the task of image inpainting, in which some missing pixels or an occluded region of an unobserved image needs to be restored. Our model can restore the occluded region by first inferring the latent variables of the incomplete image and then generating a complete image from the inferred latent variables. No OT correction is needed after the model is trained, therefore, the inference for inpainting purpose is directly performed via the short-run Langevin dynamics. Different from the MCMC inference performed on a complete image, where the gradient in the Langevin step is computed by summing over all pixels of the image, the inference performed on an occluded image computes the Langevin gradient by summing over only the visible pixels of the image.

We demonstrate the effectiveness of our model for image inpainting on the CelebA dataset, where images are occluded by different kinds of masks with random locations, including two-region mask with two randomly placed 16×16 patches, single region mask with a size of 32×32 pixels, single region mask with a size of 45×45 pixels, and three types of salt-and-pepper masks that cover 50%, 70% and 90%, respectively, of pixels of an image by randomly placed 3×3 patches. Fig. 6 displays some

qualitative results obtained by our model trained in section 7.3 for image generation. The inference step used in inpainting follows the same number of Langevin steps and the same step size as those used in the training stage. In each panel, the first row shows the original images, the second and the third rows show the occluded images and the corresponding restored images. In Fig. 7, we show that the inpainting algorithm can restore the occluded region of an image with diverse and reasonable patterns, which means that the learned model can capture a meaningful latent space of the data and the short-run MCMC inference step can traverse between different modes in the learned latent space.

7.5 Anomaly detection

Anomaly detection is another task that can help evaluate the proposed model. With a well-learned model from the normal data, we can detect the anomalous data by firstly sampling the latent code z of the given testing image $\mathbf{I}$ from the conditional distribution $q_\theta(z^K|\mathbf{I})$ by the short-run Langevin dynamics, and then computing the logarithm of the joint probability $\log p_\theta(\mathbf{I}, z)$ in Eq. (7). Based on our theory, the joint probability should be high for the normal images and low for the anomalous ones.

In the following experiments, we treat each class in the MNIST dataset as an anomalous class and leave the others as normal. We follow the protocols as in [12], [84], [85] and train the model only with the normal data. Then the model is tested with both the normal and anomalous data. To evaluate the performance, we use $\log p_\theta(\mathbf{I}, z)$ as our decision function to compute the area under the precision-recall curve (AUPRC), just like [24] does. In the test stage, we run each experiment 10 times to get the mean and variance. In Table 3, we compare our method with the related models in this task, including the VAE [9], MEG [84], BiGAN-σ [85], EnGAN [86], EBM-VAE [87], LEBM [24] and ABP model [11], which can be treated as a special case of our model without the OT calibration. Besides, we also report the results of our model with different parameter α in Eq. (12). From the table, we can find that the proposed method can get much better results than those of other methods.

7.6 Influence of the number of latent dimensions

Here we show the influence of the number of dimensions of the latent space under the same architecture. We use the SVHN dataset, and set different numbers of dimensions of the latent space, e.g., 20, 40 and 64, respectively. As shown in Table 4, with more latent dimensions, we can obtain much better results in terms of both reconstruction and generation.

TABLE 4
The performances of the proposed method on SVHN dataset with the same architecture but different numbers of latent dimensions. (Smaller is better for MSE and FID.)

# Dimension	MSE	FID
20	0.011	36.32
40	0.008	24.73
64	0.002	19.48

7.7 Ablation study

Now we explore the performances of the proposed model under different values of the parameter α introduced in Eq. (12), different step sizes of the Langevin dynamics (the s of Eq. (9)), different numbers of Langevin steps (K in Eq. (9)) and different numbers of iterations for the learning step that seeks to maximize the joint probability in Eq. (7) using the paired data $\{(\hat{z}_i, \mathbf{I}_i)\}$.

The influence of α. Firstly, we investigate the influence of α in Eq. (12), which is shown in Fig. 8. In the left subfigure, we show the OT cost from $\{\hat{z}_i\}$ to $\{z_j\}$, which serves as a distance between the $q_\theta(z^K)$ through the short-run Langevin dynamics and the prior distribution $p(z)$. It is obvious that a larger α can pull the marginal distribution $q_\theta(z^K)$ more quickly toward the prior distribution. The subfigure in the middle suggests that to get a smaller MSE loss, it is better to choose a smaller α. According to the right subfigure, we get the best FID with a medium α, namely $\alpha = 0.5$. Thus, to balance the OT cost, MSE loss and the FID, we set $\alpha = 0.5$ in the following experiments. From the curves, we also find that as the algorithm progresses, the marginal distribution $q_\theta(z^K)$ gets increasingly close to the prior distribution $p_0(z)$, and the qualities of both the reconstructed images and the generated images also increase.

TABLE 5
The influence of the step size of the Langevin dynamics.

		s=3e-3	s=1.5e-2	s=3e-2	s=6e-2
MSE	Before	0.007	0.008	0.011	0.027
	After	0.018	0.013	0.013	0.027
FID	Before	44.51	28.10	22.70	109.97
	After	40.61	26.86	21.89	87.77

The influence of the Langevin step size. Next, we show the performances of our model with different Langevin step sizes (s in Eq. (9)) in Table 5, where "Before" means that we use the model before the OT correction, and "After" means we use the trained model after the OT correction. With a small s, the MSE loss is indeed very small, but the FID is relatively large, meaning that the quality of the generated images is not very good. When s is large, e.g., $s = 6e^{-2}$ in the last column, both the MSE loss and the FID are large, which means that we cannot even get high quality reconstructed images. In this situation, the model actually doesn't converge very well. Only with the appropriate Langevin step size (in this experiment, $s = 3e^{-2}$), we can obtain a good balance between the MSE and the FID for satisfying reconstruction and generation results.

The influence of the number of Langevin steps. The number of Langevin steps K in Eq. (9) is another key factor that influences the performance of the proposed method. Theoretically, larger K will give us a more convergent MCMC inference, so as to help us get more accurate latent variables. To prove this point, we set $K = 30, 50, 100$ respectively, and keep the other parameters fixed. The results are shown in Table 6. Indeed, a larger K gives us a better result. However, a large K will also increase the running time for the whole pipeline linearly. Thus, to get a good balance between the running time and the performance, we need to choose the suitable K for different datasets.

The influence of the number of iterations inside the learning step. In Alg. 1, we actually run several iterations, denoted by L_2, of gradient ascent inside the learning step to maximize the joint probability in Eq. (7) by the paired data $\{(\hat{z}_i, \mathbf{I}_i)\}$. The results are shown in Table 7. From the table we can find that by increasing L_2, we can get much better performances for image reconstruction and generation.

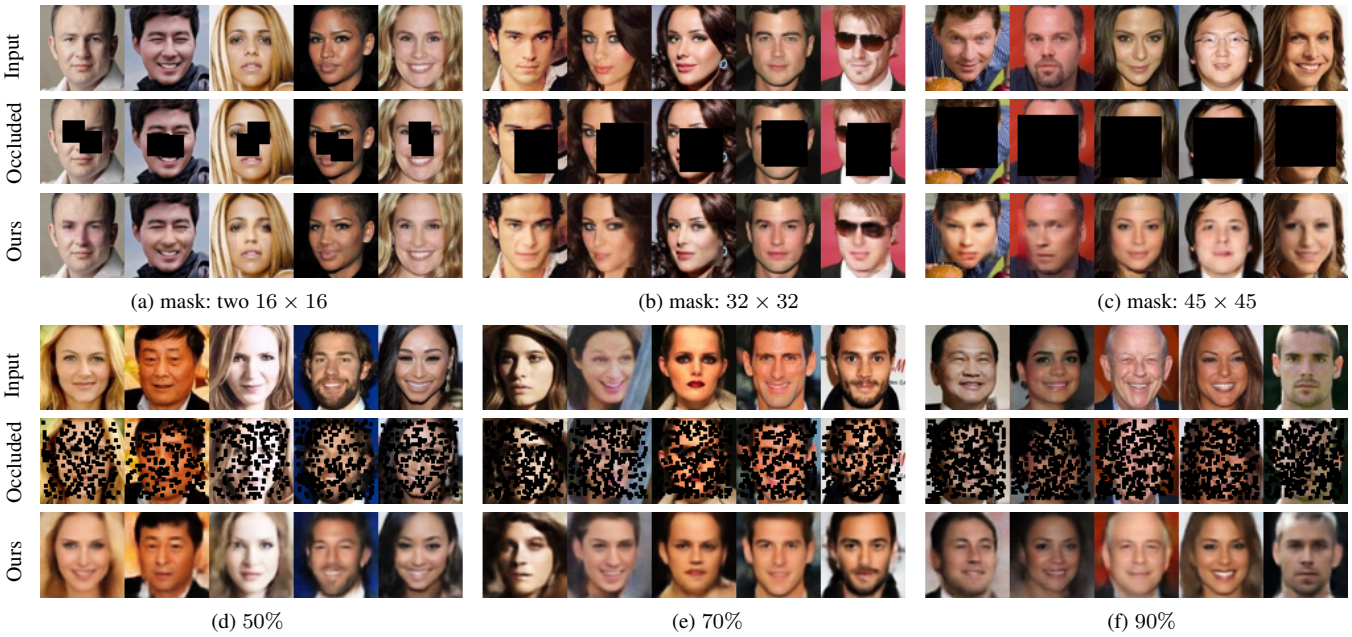


Fig. 6. The qualitative results of image inpainting for different types and levels of occlusions. In each panel, the first row shows the original images, and the second row shows the corresponding occluded images with different sizes or percentages of masks, and the third row shows the reconstructed images by our method. The mask sizes in panels (a)(b)(c) are 16×16, 32×32, and 45×45, respectively. The occlusion percentages of salt-and-pepper masks in panels (d)(e)(f) are 50%, 70%, and 90%, respectively.

Fig. 7. Diverse inpainting results of the same masked input image. Each row shows one example, where the first column shows the original image, the second column shows the masked images that need to be recovered, and the rest columns show the different inpainting results of the same masked input image in the second column. The mask size is 45×45, and the image size is 64×64.

TABLE 6
The influence of the number of Langevin steps K.

	K=30	K=50	K=100
MSE	0.014	0.011	0.007
FID	22.32	18.57	15.43

TABLE 7
The influence of the number of learning iterations.

	L_2=1	L_2=2	L_2=3
MSE	0.013	0.010	0.008
FID	21.89	17.32	14.28

7.8 Learning from incomplete data

In Section 7.4, we have shown that the model trained on fully observed images can perform image inpainting in testing stage. This is a supervised setting of image inpainting because complete data are provided in the training stage. As to an unsupervised setting, only incomplete training images are provided to learn how to restore occluded images, which is more challenging than the supervised setting.

We have shown that in Section 6 our top-down generative model with a short-run MCMC inference can learn from incomplete data, e.g., images with occluded pixels. To demonstrate this ability, we experiment on the occluded images from the CelebA dataset [76], in which images are occluded with different types of masks. To construct the dataset, we randomly select 10,000 images from the CelebA dataset, and then randomly place an occluding mask to each image. Similar to [11], we use two types of masks: single region mask and salt and pepper mask. We use different sizes of single regions, e.g., 15×15, 20×20, 25×25, 30×30, 35×35, 40×40, and 45×45, and different occlusion percentages of salt-and-pepper masks, e.g., 50%, 60%, 70%, 75%, 80%, 85%, and 90%. We compare our method with two related baselines, the VAE method

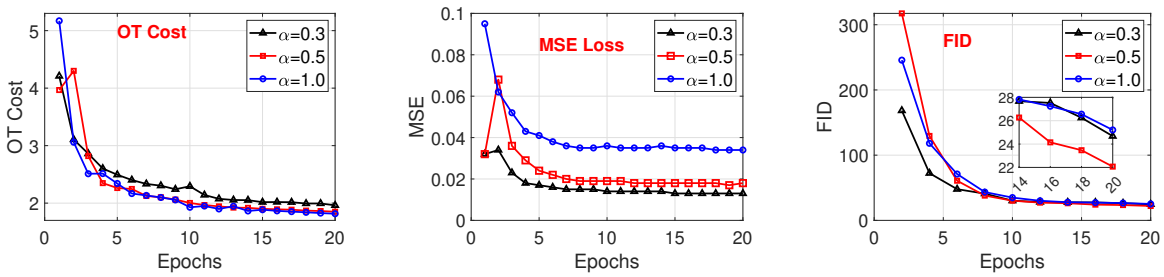


Fig. 8. The influences of α on the OT cost, MSE loss and FID over different epochs for the MNIST dataset [74].

(i.e., the generative model with variational inference) [9] and the ABP method (i.e., the generative model with an MCMC inference without using OT correction) [11], in the task of unsupervised image inpainting. We adapt the original VAE and ABP algorithms to this task by modifying their loss terms so that they are only computed on the unoccluded pixels.

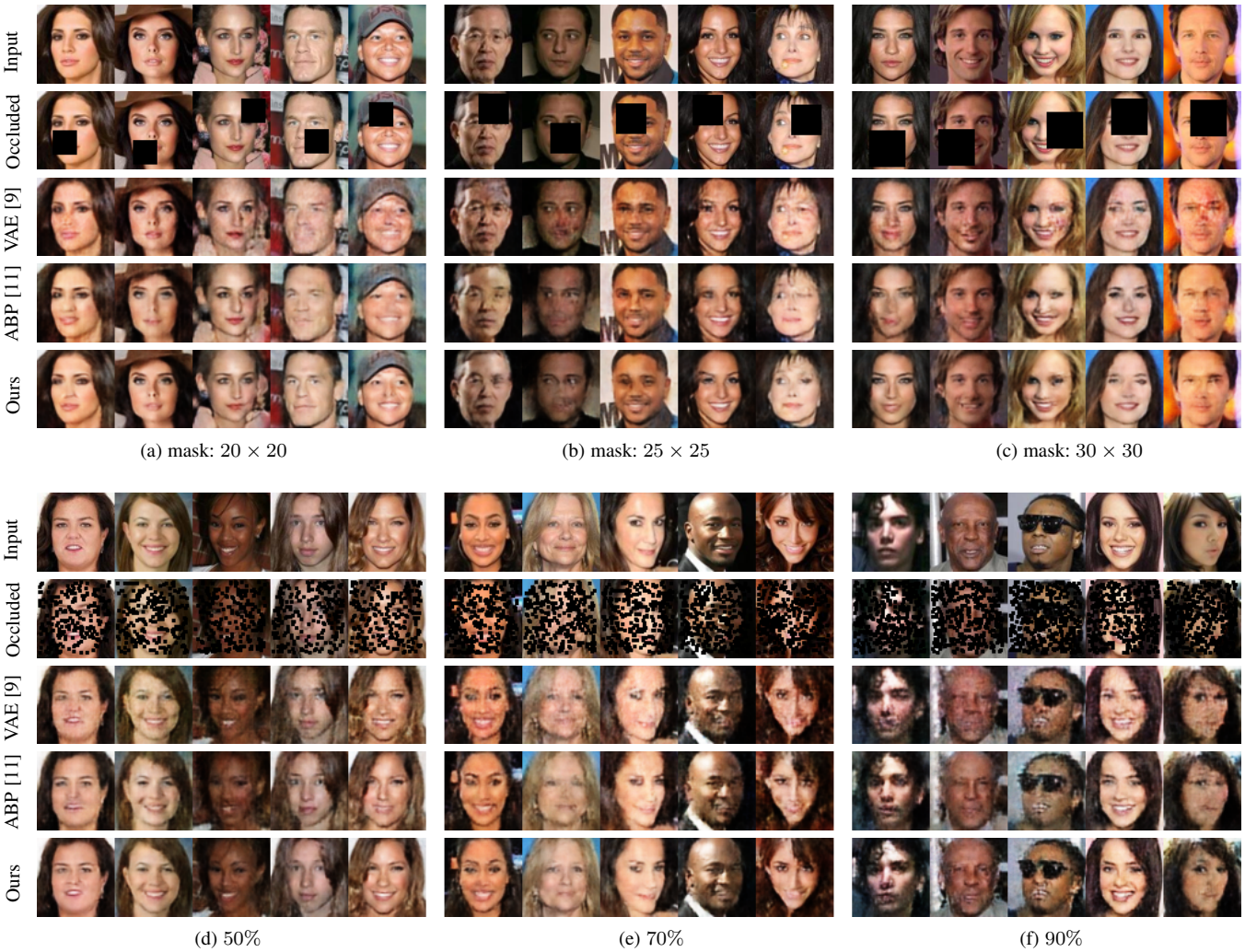
Fig. 9 shows a comparison of qualitative results, where the first row shows original images that are unknown in the training algorithm, the second row shows the corresponding occluded training images. The third, fourth, and fifth rows show the corresponding recovered images obtained by the VAE [9], the ABP [11], and our recovering algorithm presented in Alg. 3, respectively. Moreover, we measure the performance of the proposed method and the baselines by using the recovery errors calculated by the average per pixel difference between the original image and the recovered image on the occluded pixels. Table 8 presents a comparison among them in the tasks of image recovery, with different types and different levels of occlusions. For all the methods, the recovery error increases as the occlusion level (i.e., mask size or occlusion percentage) increases. Our method outperforms the baselines in all settings in terms of recovery error. Besides, in Fig. 10 we show the evolution of the MSE loss and the OT cost over epochs for the recovery task, where the images are occluded by one single 20×20 mask. From the figure we can see that both the MSE loss and the OT cost decrease consistently as the recovery algorithm proceeds.

8 Conclusion

Learning generative models is a fundamental problem in computer vision and machine learning. In this paper, we put emphasis on learning top-down generative models by maximum likelihood estimation, in which the inference is accomplished by an efficient but biased short-run MCMC, such as Langevin dynamics. We propose to use the optimal transport (OT) theory to correct the bias of the short-run MCMC-based inference in training the deep top-down generative models. Specifically, in each iteration, we correct the bias of the marginal distribution of the latent variables inferred by the short-run Langevin dynamics through the OT map between this distribution and the prior distribution. We explicitly transport the biased inferred vectors to the prior distribution to enforce the aggregated inference distribution to be the prior distribution. In such a way, the distribution of the inferred latent vector will finally converge to the prior distribution, thus improving the accuracy of the subsequent model parameter learning. Experimental results show that the proposed training method performs better than the models using MCMC inference without OT correction and the models using variational inference on the tasks like image reconstruction, image generation, supervised image inpainting, anomaly detection, and unsupervised image recovery.

References

[1] J. Xie, Y. Lu, S.-C. Zhu, and Y. N. Wu, "A theory of generative convnet," in *International Conference on Machine Learning (ICML)*, 2016.
[2] S. C. Zhu, Y. N. Wu, and D. Mumford, "Minimax entropy principle and its application to texture modeling," *Neural Computation*, vol. 9, no. 8, pp. 1627–1660, 1997.
[3] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*. Springer Science & Business Media, 2008.
[4] A. Barbu and S.-C. Zhu, *Monte Carlo Methods*. Springer, 2020.
[5] S.-C. Zhu, "Statistical modeling and conceptualization of visual patterns," *IEEE transactions on pattern analysis and machine intelligence*, vol. 25, no. 6, pp. 691–712, 2003.
[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 2672–2680.
[7] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2016.
[8] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning (ICML)*, 2017, pp. 214–223.
[9] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations (ICLR)*, 2013.
[10] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *International Conference on Machine Learning (ICML)*, vol. 32, 2014, pp. 1278–1286.
[11] T. Han, Y. Lu, S.-C. Zhu, and Y. N. Wu, "Alternating back-propagation for generator network," in *The AAAI Conference on Artificial Intelligence (AAAI)*, 2017, pp. 1976–1984.
[12] E. Nijkamp, B. Pang, T. Han, L. Zhou, S.-C. Zhu, and Y. N. Wu, "Learning multi-layer latent variable model via variational optimization of short run MCMC for approximate inference," in *European Conference on Computer Vision (ECCV)*, 2020, pp. 361–378.
[13] H. Ledig, L. Theis, F. Huszan, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 105–114.
[14] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5505–5514.
[15] Y.-J. Li, Y.-C. Chen, Y.-Y. Lin, X. Du, and Y.-C. F. Wang, "Recover and identify: A generative dual model for cross-resolution person re-identification," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 8089–8098.
[16] C. Zheng, T. Cham, and J. Cai, "Pluralistic image completion," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1438–1447.
[17] J. Gu, Y. Shen, and B. Zhou, "Image processing using multi-code GAN prior," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 3009–3018.
[18] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin, "Variational autoencoder for deep learning of images, labels and captions," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 2352–2360.



(a) mask: 20×20 (b) mask: 25×25 (c) mask: 30×30

(d) 50% (e) 70% (f) 90%

Fig. 9. A comparison of recovery results by different methods in unsupervised recovery tasks with different levels of occlusions. In each panel, the first row shows some original images that are unobserved in all training algorithms, the second row shows the corresponding occluded images with different sizes of masks or percentages of noises, and the third, fourth and fifth rows show the reconstructed images by the VAE model [9], the ABP model [11] and our model, respectively. The mask sizes in panels (a)(b)(c) are 20×20, 25×25, and 30×30, respectively. The occlusion percentages of salt-and-pepper masks in panels (d)(e)(f) are 50%, 70%, and 90%, respectively. The input images are of size 64×64.

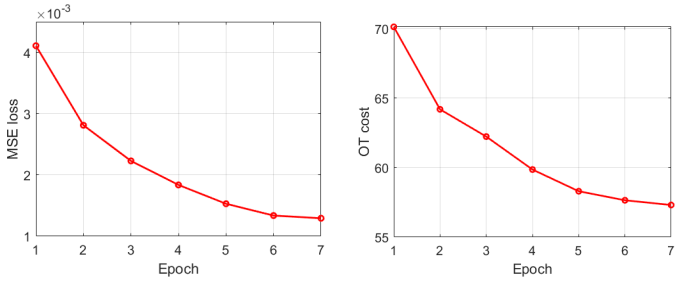


Fig. 10. The changes of the MSE loss and OT cost over epochs in the unsupervised recovery task with a single 20×20 block mask.

[19] T. Nguyen-Phuoc, C. Li, L. Theis, C. Richardt, and Y.-L. Yang, "HoloGAN: Unsupervised learning of 3D representations from natural images," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 7587–7596.

[20] N. Skafte and S. r. Hauberg, "Explicit disentanglement of appearance and perspective in generative models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 1016–1026.

[21] T. Aumentado-Armstrong, S. Tsogkas, A. Jepson, and S. Dickinson, "Geometric disentanglement for generative latent shape models," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 8180–8189.

[22] S. N, B. Paige, J.-W. van de Meent, A. Desmaison, N. Goodman, P. Kohli, F. Wood, and P. Torr, "Learning disentangled representations with semi-supervised deep generative models," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 5925–5935.

[23] S. Ren, D. Li, Z. Zhou, and P. Li, "Estimate the implicit likelihoods of GANs with application to anomaly detection," in *Proceedings of The Web Conference 2020 (WWW)*, 2020, pp. 2287–2297.

[24] B. Pang, T. Han, E. Nijkamp, S.-C. Zhu, and Y. N. Wu, "Learning latent space energy-based prior model," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[25] Y. Zhu, M. Elhoseiny, B. Liu, X. Peng, and A. Elgammal, "A generative adversarial approach for zero-shot learning from noisy texts," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1004–1013.

[26] Y. Zhu, J. Xie, B. Liu, and A. Elgammal, "Learning feature-to-feature translator by alternating back-propagation for generative zero-shot learning," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 9843–9853.

[27] J. Zhang, J. Xie, and N. Barnes, "Learning noise-aware encoder-decoder from noisy labels by alternating back-propagation for saliency detection," in *European Conference on Computer Vision (ECCV)*, 2020, pp. 349–366.

[28] J. Zhang, D. Fan, Y. Dai, S. Anwar, F. S. Saleh, T. Zhang, and N. Barnes, "Uc-net: Uncertainty inspired RGB-D saliency detection via conditional variational autoencoders," in *IEEE Conference on Computer Vision and*

TABLE 8
A comparison of performance among the VAE model [9], the ABP model [11] and our model for unsupervised image recovery, with different types and percentages of occlusions.

	single region mask						
Mask size	15×15	20×20	25×25	30×30	35×35	40×40	45×45
VAE [9]	0.0020	0.0024	0.0035	0.0054	0.0197	0.0204	0.0409
ABP [11]	**0.0016**	0.0020	0.0025	0.0032	0.0054	0.00107	0.0196
Ours	**0.0016**	**0.0019**	**0.0024**	**0.0029**	**0.0048**	**0.0096**	**0.0164**

	Salt-and-pepper mask						
Occlusion percentage	50%	60%	70%	75%	80%	85%	90%
VAE [9]	0.0027	0.0027	0.0035	0.0036	0.0037	0.0038	0.0049
ABP [11]	0.0022	0.0025	0.0026	0.0030	**0.0030**	0.0032	0.0035
Ours	**0.0021**	**0.0024**	**0.0026**	**0.0029**	**0.0030**	**0.0031**	**0.0034**

Pattern Recognition (CVPR), 2020, pp. 8579–8588.

[29] S. Tulyakov, M. Liu, X. Yang, and J. Kautz, "Mocogan: Decomposing motion and content for video generation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1526–1535.

[30] J. Xie, R. Gao, Z. Zheng, S.-C. Zhu, and Y. N. Wu, "Learning dynamic generator model by alternating back-propagation through time," in *The AAAI Conference on Artificial Intelligence (AAAI)*, vol. 33, 2019, pp. 5498–5507.

[31] ——, "Motion-based generator model: Unsupervised disentanglement of appearance, trackable and intrackable motions in dynamic patterns," in *The AAAI Conference on Artificial Intelligence (AAAI)*, 2020, pp. 12 442–12 451.

[32] Y. Zhu, M. R. Min, A. Kadav, and H. P. Graf, "S3VAE: self-supervised sequential VAE for representation disentanglement and data generation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6537–6546.

[33] P. Langevin, "On the theory of brownian motion," *Journal of Statistical Physics*, 1908.

[34] R. M. Neal, "MCMC using Hamiltonian dynamics," *Handbook of Markov Chain Monte Carlo*, vol. 2, no. 11, p. 2, 2011.

[35] C. Villani, *Optimal Transport: Old and New*. Springer Science & Business Media, 2008, vol. 338.

[36] A. Razavi, A. van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with VQ-VAE-2," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 14 837–14 847.

[37] A. Vahdat and J. Kautz, "NVAE: A deep hierarchical variational autoencoder," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[38] J. He, A. M. Lehrmann, J. Marino, G. Mori, and L. Sigal, "Probabilistic video generation using holistic attribute control," in *European Conference on Computer Vision (ECCV)*, vol. 11209, 2018, pp. 466–483.

[39] G. Yang, X. Huang, Z. Hao, M. Liu, S. J. Belongie, and B. Hariharan, "Pointflow: 3d point cloud generation with continuous normalizing flows," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 4540–4549.

[40] J. Aneja, H. Agrawal, D. Batra, and A. G. Schwing, "Sequential latent spaces for modeling the intention during diverse image captioning," in *International Conference on Computer Vision (ICCV)*, 2019, pp. 4260–4269.

[41] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning," in *International Conference on Learning Representations (ICLR)*, 2018.

[42] N. Loo, S. Swaroop, and R. E. Turner, "Generalized variational continual learning," in *International Conference on Learning Representations (ICLR)*, 2021.

[43] M. Rosca, B. Lakshminarayanan, and S. Mohamed, "Distribution matching in variational inference," *arXiv preprint arXiv:1802.06847*, 2018.

[44] M. D. Hoffman and M. J. Johnson, "ELBO surgery: yet another way to carve up the variational evidence lower bound," in *Workshop in Advances in Approximate Bayesian Inference, NIPS*, vol. 1, no. 2, 2016.

[45] T. Han, Y. Lu, J. Wu, X. Xing, and Y. N. Wu, "Learning generator networks for dynamic patterns," in *IEEE Winter Conference on Applications of Computer Vision, WACV*, 2019, pp. 809–818.

[46] T. Han, X. Xing, and Y. N. Wu, "Learning multi-view generator network for shared representation," in *International Conference on Pattern Recognition, ICPR*, 2018, pp. 2062–2068.

[47] X. Xing, R. Gao, T. Han, S.-C. Zhu, and Y. N. Wu, "Deformable generator networks: unsupervised disentanglement of appearance and geometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[48] E. Nijkamp, B. Pang, T. Han, S.-C. Zhu, and Y. N. Wu, "Learning multi-layer latent variable model via variational optimization of short run MCMC for approximate inference," in *European Conference on Computer Vision (ECCV)*, 2020.

[49] G. Peyré and M. Cuturi, "Computational optimal transport," *Found. Trends Mach. Learn.*, vol. 11, no. 5-6, pp. 355–607, 2019.

[50] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein GANs," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 5769–5779.

[51] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[52] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf, "Wasserstein auto-encoders," in *International Conference on Learning Representations (ICLR)*, 2018.

[53] A. Korotin, V. Egiazarian, A. Asadulaev, A. Safin, and E. Burnaev, "Wasserstein-2 generative networks," in *International Conference on Learning Representations (ICLR)*, 2021.

[54] B. Amos, L. Xu, and J. Z. Kolter, "Input convex neural networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 146–155.

[55] D. An, Y. Guo, N. Lei, Z. Luo, S.-T. Yau, and X. Gu, "AE-OT: A new generative model based on extended semi-discrete optimal transport," in *International Conference on Learning Representations (ICLR)*, 2020.

[56] D. An, Y. Guo, M. Zhang, X. Qi, N. Lei, and X. Gu, "AE-OT-GAN: Training GANs from data specific latent distribution," in *European Conference on Computer Vision (ECCV)*, 2020, p. 548–564.

[57] Y. Brenier, "Polar factorization and monotone rearrangement of vector-valued functions," *Comm. Pure Appl. Math.*, vol. 44, no. 4, pp. 375–417, 1991.

[58] C. Villani, *Topics in Optimal Transportation*. American Mathematical Society, 2003, vol. 58.

[59] S. Endre, *Lecture Notes on Finite Element Methods for Partial Differential Equations*. University of Oxford, 2020.

[60] J.-D. Benamou, B. D. Froese, and A. M. Oberman, "Numerical solution of the optimal transportation problem using the monge-ampère equation," *J. Comput. Phys*, 2014.

[61] D. X. Gu, F. Luo, j. Sun, and S.-T. Yau, "Variational principles for minkowski type problems, discrete optimal transport, and discrete monge-ampère equations," *Asian Journal of Mathematics*, vol. 20, no. 2, p. 383–398, 2016.

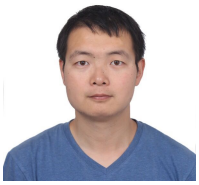
[62] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transportation distances," in *Advances in Neural Information Processing Systems (NIPS)*, 2013, pp. 2292–2300.

[63] P. Dvurechensky, A. Gasnikov, and A. Kroshnin, "Computational optimal transport: Complexity by accelerated gradient descent is better than by sinkhorn's algorithm," in *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 2018.

[64] M. Blondel, V. Seguy, and A. Rolet, "Smooth and sparse optimal transport," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[65] S. D. Marino and A. Gerolin, "Optimal transport losses and sinkhorn algorithm with general convex regularization," *arXiv preprint arXiv:2007.00976*, 2020.

[66] A. Genevay, M. Cuturi, G. Peyré, and F. Bach, "Stochastic optimization for large-scale optimal transport," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 3440–3448.
[67] V. Seguy, B. B. Damodaran, R. Flamary, N. Courty, A. Rolet, and M. Blondel, "Large-scale optimal transport and mapping estimation," in *International Conference on Learning Representations (ICLR)*, 2018.
[68] S.-C. Zhu and D. Mumford, "Grade: Gibbs reaction and diffusion equations," in *International Conference on Computer Vision (ICCV)*, 1998, pp. 847–856.
[69] Y. T. Lee and A. Sidford, "Path finding methods for linear programming: Solving linear programs in $\tilde{O}$(sqrt(rank)) iterations and faster algorithms for maximum flow," in *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2014, pp. 424–433.
[70] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, 1955.
[71] F. Aurenhammer, F. Hoffmann, and B. Aronov, "Minkowski-type theorems and least-squares clustering," *Algorithmica*, vol. 20, no. 1, pp. 61–76, 1998.
[72] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2014.
[73] Y. Nesterov, "Smooth minimization of non-smooth functions," *Mathematical Programming*, vol. 103, p. 127–152, 2005.
[74] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner., "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.
[75] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.
[76] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "From facial expression recognition to interpersonal relation prediction," *International Journal of Computer Vision (IJCV)*, vol. 126, no. 5, pp. 550–569, 2018.
[77] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a nash equilibrium," *Advances in Neural Information Processing Systems (NIPS)*, pp. 6626–6637, 2017.
[78] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010, pp. 249–256.
[79] B. Dai and D. Wipf, "Diagnosing and enhancing VAE models," in *International Conference on Learning Representations (ICLR)*, 2019.
[80] P. Ghosh, M. S. M. Sajjadi, A. Vergari, M. Black, and B. Scholkopf, "From variational to deterministic autoencoders," in *International Conference on Learning Representations (ICLR)*, 2020.
[81] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, "Ladder variational autoencoders," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 29, 2016.
[82] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," in *International Conference on Learning Representations (ICLR)*, 2016.
[83] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018.
[84] R. Kumar, A. Goyal, A. C. Courville, and Y. Bengio, "Maximum entropy generators for energy-based models," *arXiv:1901.08508*, 2019.
[85] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, "Efficient GAN-based anomaly detection," *arXiv: 1802.06222*, 2018.
[86] R. Kumar, S. Ozair, A. Goyal, A. Courville, and Y. Bengio, "Maximum entropy generators for energy-based models," *arXiv preprint arXiv:1901.08508*, 2019.
[87] T. Han, E. Nijkamp, L. Zhou, B. Pang, S.-C. Zhu, and Y. N. Wu, "Joint training of variational auto-encoder and latent energy-based model," in *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020.



Dongsheng An is currently a PhD candidate at the Department of Computer Science, Stony Brook University. Prior to that, he received his M.S. and B.S. from Tsinghua University, China. His main research interests include computational optimal transport, deep generative modeling and computational conformal/quasi-conformal geometry.

Jianwen Xie received his Ph.D. degree in statistics from University of California, Los Angeles (UCLA) in 2016. He is currently a senior research scientist at Baidu Research USA. Before joining Baidu, he was a senior research scientist at Hikvision Research Institute USA from 2017 to 2020, and a staff research associate and postdoctoral researcher in the Center for Vision, Cognition, Learning, and Autonomy (VCLA) at UCLA from 2016 to 2017. His research interests focus on generative modeling and learning with applications in computer vision.

Ping Li received his Ph.D. in Statistics in 2007, from Stanford University, where he also earned a master's degree in Computer Science and a master's degree in Eletrical Engineering. Prior to Stanford, Ping Li gradated two master's degrees from the University of Washington (Seattle). Ping Li was a recipient of the Young Instigator Award from the Office of Naval Research (ONR-YIP) and a recipient of the Young Investigator Award from the Air Force Office of Scientific Research (AFOSR-YIP).