Learning Deep Latent Variable Models by Short-Run MCMC Inference with Optimal Transport Correction

Dongsheng An, Jianwen Xie, Ping Li
Cognitive Computing Lab
Baidu Research
10900 NE 8th St. Bellevue, WA 98004, USA
{dongshengan15, jianwen.kenny, pingli98}@gmail.com

Abstract

Learning latent variable models with deep top-down architectures typically requires inferring the latent variables for each training example based on the posterior distribution of these latent variables. The inference step typically relies on either time-consuming long-run Markov chain Monte Carlo (MCMC) sampling or a separate inference model for variational learning. In this paper, we propose to use a short-run MCMC, such as a short-run Langevin dynamics, as an approximate flow-based inference engine. The bias existing in the output distribution of the non-convergent short-run Langevin dynamics is corrected by the optimal transport (OT), which aims at transforming the biased distribution produced by the finite-step MCMC to the prior distribution with a minimum transport cost. Our experiments not only verify the effectiveness of the OT correction for the short-run MCMC, but also demonstrate that the latent variable model trained by the proposed strategy performs better than the variational auto-encoder (VAE) in terms of image reconstruction/generation and anomaly detection.

1. Introduction

Recent years have seen a great success of deep generative models in numerous computer vision applications, such as image generation [10, 16, 13], image recovery [21, 12, 23], image representation [33, 29], image disentanglement [35, 4, 25], anomaly detection [34, 31], etc. Such models typically include simple and expressive generator networks, which are latent variable models assuming that each observed example is generated by a low-dimensional vector of latent variables, and the latent vector follows a non-informative prior distribution, such as Gaussian distribution. Since high dimensional visual data (e.g., images) usually lie on low-dimensional manifolds embedded in the high-dimensional space, learning latent variable models of visual data is of fundamental importance in the field of computer vision for the sake of unsupervised representation learning. The challenge mainly comes from the inference of the latent variables for each observation, which typically relies on Markov chain Monte Carlo (MCMC) [24, 6] 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 posterior distribution of the latent variables is parameterized by a highly non-linear deep neural network, the MCMC-based inference can suffer from non-convergence and inefficiency problems, thus affecting the accuracy of the model parameter estimation.

To avoid inefficient MCMC sampling from the posterior, variational inference [16] becomes an attractive alternative by approximating the intractable posterior via a tractable network. Despite the growing prevalence and popularity of the variational auto-encoder (VAE) [16], its drawbacks are increasingly obvious. (i) It parameterizes the intrinsic iterative inference process by an extrinsic feedforward inference model. These extra parameters due to the reparameterization have to be estimated together with those of the generator network. (ii) Such a joint training 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. Only when the Kullback-Leibler (KL)-divergence between the inference and the posterior distribution is equal to zero, the variational inference is equivalent to the desired maximum likelihood estimation. This goal is usually infeasible in practice. (iii) An extra effort is required to made in designing the inference model of VAE, especially for the generators that have complicated dependency structures with the latent variables, e.g., [30] proposed a top-down generator with multiple layers of latent variables, [39, 40] proposed dynamic generators with time sequences of latent variables. It is not a simple task to design inference models that infer latent variables for models mentioned above. An arbitrary design of the inference model cannot guarantee the performance.

In this paper, we will totally abandon the idea of reparameterizing the inference process, and reuse the MCMC-based inference for training deep latent variable models. To be specific, we use a short-run MCMC, such as a short-run Langevin dynamics [19, 26], to perform the inference of the latent vectors during training. 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 and propose to use the optimal transport (OT) method [38] 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 latent variable model. We correct the bias of the short-run MCMC by performing an optimal transport from the result distribution produced by the short-run MCMC to the prior distribution. This operation is to minimize the OT cost between the inference distribution and the prior distribution, in which we don't optimize any parameters in the flow model but update its output. With the corrected inference output, we can update the parameters of the latent variable model more accurately.

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

There are several advantages in the proposed algorithm: (i) efficiency: The learning and inference of the model are efficient with a short-run MCMC. (ii) convenience: The approximate inference model represented by the short-run MCMC is automatic 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. (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 deep latent variable model by a non-convergent short-run MCMC inference with OT correction. (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 latent variable models.

2. Related work

Variational inference. VAE [16] is a popular method to learn generator network by simultaneously training a tractable inference network to approximate the intractable posterior distribution of the latent variables. 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 Langevin sampling from the posterior distribution, followed by an optimal transport correction.

Alternating back-propagation algorithm. The maximum likelihood learning of the generator network, including its dynamic version, can be achieved by the alternating back-propagation (ABP) algorithm [13, 39], 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 by Langevin sampling from the posterior distribution, and (ii) learning step: updating the model parameters based on the training data and the inferred latent variables by gradient descent. Both steps compute the gradients with the help of back-propagation. The ABP algorithm has been successfully applied to saliency detection [43], zero-shot learning [46], and disentangled representation learning [41, 40], etc.

Optimal Transport. Optimal transport (OT) is used to compute the distance between two measures and is able to push forward the source distribution to the target distribution [38, 32]. Recently, OT has been widely used in the generative models to help generate high quality samples. For example, by replacing the original KL-divergence in the GAN models [10] with the W_1 distance, Arjovsky et al. [3] proposed the WGAN model to achieve better convergence and generate higher quality samples. Tolstikhin et al. [36] proposed the Wasserstein variational auto-encoder that minimizes the Wasserstein distance between the inference model and the posterior distribution. 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 [1, 2].

3. Maximum likelihood learning of deep latent variable model

Let $\mathbf{I}$ be a D-dimensional observed data example, such as an image. Let z be the d-dimensional vector of continuous latent variables. 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 $z \sim \mathcal{N}(0, I_d)$. I_d and I_D are d-dimensional and D-dimensional identity matrices, respectively. We assume $d \ll D$. The generator network is essentially a non-linear latent variable 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 $p_\theta(z|\mathbf{I}) = p_\theta(\mathbf{I}, z)/p_\theta(\mathbf{I})$.

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 the minimization of $\text{KL}(p_{\text{data}}||p_\theta)$ when the number of training examples n is large enough [13].

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

$$\theta_{t+1} = \theta_t + \gamma_t \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta \log p_\theta(\mathbf{I}_i), \tag{3}$$

where γ_t is the learning rate depending on time t and the gradient of the log probability is given by

$$\begin{aligned} \nabla_\theta \log p_\theta(\mathbf{I}) &= \frac{1}{p_\theta(\mathbf{I})}\nabla_\theta p_\theta(\mathbf{I}) \\ &= \int [\nabla_\theta \log p_\theta(\mathbf{I}, z)] \frac{p_\theta(\mathbf{I}, z)}{p_\theta(\mathbf{I})} dz \\ &= \text{E}_{p_\theta(z|\mathbf{I})}[\nabla_\theta \log p_\theta(\mathbf{I}, z)]. \end{aligned} \tag{4}$$

To compute $\nabla_\theta \log p_\theta(\mathbf{I})$ in Eq. (4), we need to estimate $\nabla_\theta \log p_\theta(\mathbf{I}, z)$. According to Eq. (1), the logarithm of the join distribution is given by

$$\log p_\theta(\mathbf{I}, z) = -\frac{1}{2\sigma^2}\|\mathbf{I} - g_\theta(z)\|^2 - \frac{1}{2}\|z\|^2 + \text{const}, \tag{5}$$

where the constant term is independent of z or 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. (3), the key is to compute the intractable expectation term in Eq. (4), 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 [19, 45], which is a gradient-based MCMC method, can produce samples from the posterior density $p_\theta(z|\mathbf{I})$ by recursively computing

$$z^{k+1} = z^k + \frac{s^2}{2}\nabla_z \log p_\theta(z|\mathbf{I}) + s\xi_k, \tag{6}$$

where k indexes the time step of Langevin dynamics, $\xi_k \sim \mathcal{N}(0, I_d)$ is a random noise diffusion. 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 $p_\theta(z|\mathbf{I})$.

4.2. Short-run Langevin dynamics

It is not sensible or realistic to use a long-run MCMC to train the model. Within each iteration, running a finite number of Langevin steps for inference toward $p_\theta(z|\mathbf{I})$ appears to be practical. Thus, a short-run K-step Langevin dynamics is given by

$$\begin{aligned} &z^0 \sim p_0(z), \\ &z^{k+1} = z^k + \frac{s^2}{2}\nabla_z \log p_\theta(z|\mathbf{I}) + s\xi_k, k = 1, .., K. \end{aligned} \tag{7}$$

The initial distribution p_0 is assumed to be the Gaussian distribution in this paper. Following [30], 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

$$q_\theta(z^K|\mathbf{I}) = \int p_0(z^0)\kappa_\theta(z^K|z^0, \mathbf{I})dz^0, \tag{8}$$

and the corresponding marginal distribution of z^K is

$$q_\theta(z^K) = \int q_\theta(z^K|\mathbf{I})p_{data}(\mathbf{I})d\mathbf{I}. \tag{9}$$

If the MCMC converges, $q_\theta(z^K)$ should be close to the prior distribution $p(z)$, otherwise there is a gap between them.

Eq. (7) 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. (8), it might not converge to the true posterior distribution $p_\theta(z|\mathbf{I})$. [30] 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 $p_\theta(z|\mathbf{I})$.

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., $\mathrm{KL}(q_\theta(z^K|\mathbf{I})|p_\theta(z|\mathbf{I}))$, we use OT to minimize the transport cost between the marginal 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 latent variable model $\mathbf{I} = g_\theta(z)$ that generates an observed image $\mathbf{I}$ from a latent vector z, we iterate the following three steps. (i) Inference step: we first infer the latent vector for each observed image $\mathbf{I}_i$ by a K-step short-run MCMC, i.e., $\hat{z} \sim p_\theta(z^K|\mathbf{I}_i)$, and then we obtain a population $\{\hat{z}_i\}$ of the inferred latent vectors for all observed data $\{\mathbf{I}_i\}$, where $\{\hat{z}_i\} \sim q_\theta(z^K)$; (ii) 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; (iii) Learning step: Given the observed images and their corresponding inferred latent vectors, we update θ by following Eq. (3) and Eq. (4). 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.

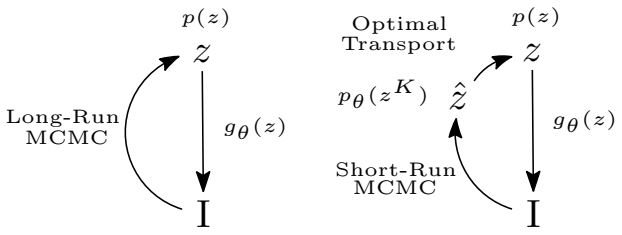


Figure 1. Diagrams of two learning strategies for latent variable models: (left) the long-run MCMC inference framework. (right) the proposed framework using a short-run MCMC with OT correction.

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.,

$$\hat{z}_i \leftarrow \alpha T(\hat{z}_i) + (1-\alpha)\hat{z}_i, \tag{10}$$

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 [13]. 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 $\mathcal{N}(0, I_d)$, and (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. (7). 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. (3) and Eq. (4) 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 [37], and then solve the following assignment problem:

$$\min_{\pi \in \Pi} \sum_{i,j=1}^{n} \pi_{ij} c_{ij} \tag{11}$$

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: $T : \hat{z}_i \to z_j$ 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 [22]). Similarly, the classical Hungarian algorithm [17] 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 [7], 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. (11). Here we extend the original dual formula for the semi-discrete OT in [5, 11, 1] to the following minimization problem in our discrete setting:

$$\min_h E(h) = \frac{1}{n} \sum_{j=1}^{n} \max_j \{\langle \hat{z}_i, z_j \rangle + h_j\} - \frac{1}{n} \sum_{j=1}^{n} h_j. \tag{12}$$

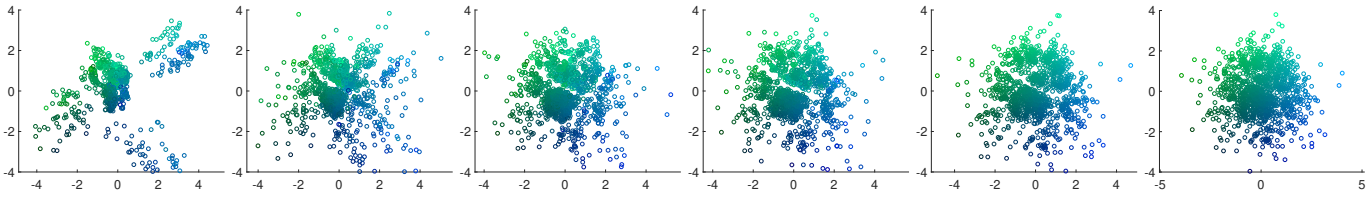
The above problem is convex as it is the maximum of the summation of n hyperplanes. Thus, it can be solved by the gradient descent algorithm. The gradient is computed by $\frac{\partial E}{\partial h_j} = \frac{\#J_j}{n} - \frac{1}{n}$, where $J_j = \{i | \langle \hat{z}_i, z_j \rangle + h_j \geq \langle \hat{z}_i, z_k \rangle + h_k \; \forall \, k \in [n]\}$ 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 [15].

Since Eq. (12) 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 $T : \hat{z}_i \to z_j, j = \arg\max_k \langle \hat{z}_i, z_k \rangle + h_k^* \; \forall \, k \in [n]$. 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. (12) is $O(n^2/\sqrt{\epsilon})$ [27]. 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.

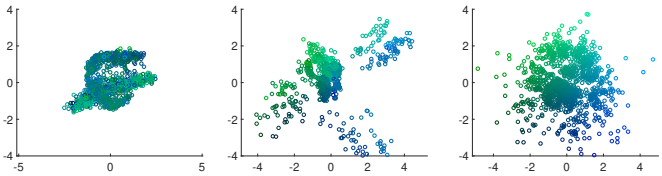
6. Experiments

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, and (iii) successfully perform anomaly detection. To show the performance of our method, we experiment on MNIST [20], SVHN [28] and CelebA [44] datasets. Details



(a) Iter 1 (b) Iter 2 (c) Iter 4 (d) Iter 6 (e) Iter 8 (f) Prior distribution

Figure 2. Visualization of the latent codes sampled from the marginal distribution $q_\theta(z^K)$ at different iterations and the prior distribution



(a) VAE [16] (b) ABP [13] (c) Ours

Figure 3. The output marginal distributions of z by different models trained on images from classes "0" and class "1" of MNIST dataset.

about the design of the generator architecture, the choices of the model hyperparameters and the optimization method for each dataset can be found in the supplementary material. 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 [14] to measure the quality of the reconstructed and generated images.

6.1. Latent space analysis

To verify that the proposed method does correct the short-run marginal distribution $q_\theta(z^K)$ of the latent variables, we pick up the classes "0" and "1" of the MNIST dataset, from which we learn our model with the latent space dimension set to be 2 for better visualization. We first show the evolution of $q_\theta(z^K)$ at different iterations of our model in Fig. 2, where the iteration indicates the number of OT corrections. From 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. Fig. 3 also shows a comparison of the latent vectors inferred by the VAE model [16], the ABP model [13] and our model, respectively. The distributions of latent vectors inferred by the VAE and the ABP models are far from the prior (Gaussian) distribution, while the marginal distribution $q_\theta(z^K)$ of our model looks much closer to it.

6.2. Image modeling

We evaluate the quality of both the reconstructed and generated images. With a well-learned model, the marginal distribution of $q_\theta(z^K)$ should match the prior distribution well. In such a case, the generator will be a probability transformation from the prior Gaussian distribution to the image distribution, and we can synthesize a high quality image by $\mathbf{I} = g_\theta(z)$ with a latent vector z sampled from the prior distribution. Additionally, the model can be useful for reconstruction. In the following, we compare our model to the VAE [16], its variants 2sVAE [8] and RAE [9]. We also compare with the ABP model [13] and its variant SRI [30], whose generator has multiple layers of latent variables. The last model we compare is the LEBM model [31], which uses an energy-based short-run MCMC to infer the latent variables of each observed image.

In Fig. 4, we show both the reconstructed and the gener-

Figure 4. The reconstructed (the first column) and the generated images (the second column) of MNIST [20] (the first row), SVHN [28] (the second row) and CelebA [44] (the third row) datasets.

Models		VAE	2sVAE	RAE	ABP	SRI	SRI (L=5)	LEBM	Ours
MNIST	MSE	0.023	0.026	0.015	-	0.019	0.015	-	**0.0008**
	FID	19.21	18.81	23.92	-	-	-	-	**14.28**
SVHN	MSE	0.019	0.019	0.014	-	0.018	0.011	0.008	**0.002**
	FID	46.78	42.81	40.02	49.71	44.86	35.23	29.44	**19.48**
CelebA	MSE	0.021	0.021	0.018	-	0.020	0.015	0.013	**0.010**
	FID	65.75	49.70	40.95	51.50	61.03	47.95	37.87	**29.75**

Table 1. 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.

Heldout Digit	1	4	5	7	9
VAE	0.063	0.337	0.325	0.148	0.104
MEG	0.281± 0.035	0.401± 0.061	0.402± 0.062	0.290± 0.040	0.342± 0.034
Bigan-σ	0.287± 0.023	0.443± 0.029	0.514± 0.029	0.347± 0.017	0.307± 0.028
LEBM	0.336± 0.008	0.630± 0.017	0.619± 0.013	0.463± 0.009	0.413± 0.010
ABP	0.095± 0.028	0.138± 0.037	0.147±0.026	0.138± 0.021	0.102±0.033
Ours	**0.353±0.021**	**0.770± 0.024**	**0.726±0.030**	**0.550±0.013**	**0.555±0.023**

Table 2. AUPRC scores (larger is better) for unsupervised anomaly detection on the MNIST dataset. Numbers are taken from [31] and results for our model are averaged over 10 experiments for variance.

ated 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 1, we use the MSE to test the quality of the reconstructed images and the FID score to quantify the quality the generated images. From the table we can find that the proposed method outperforms the other methods in the tasks of reconstruction and generation.

6.3. 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. (5). 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 [18, 42, 30] 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 [31] does. In the test stage, we run each experiment 10 times to get the mean and variance. In Table 2, we compare our method with the related models in this task, including the VAE [16], MEG [18], BiGAN-σ [42], LEBM [31] and ABP model [13], which can be treated as a special case of our model without the OT calibration. From the table, we can find that the proposed method can get much better results than those of other methods.

6.4. 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 3, with more latent dimensions, we can obtain much better results in terms of both reconstruction and generation.

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

Table 3. 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.)

6.5. Ablation study

Now we explore the performances of the proposed model under different values of the parameter α introduced in Eq. (10), different step sizes of the Langevin dynamics (the s of Eq. (7)), different numbers of Langevin steps (K in Eq. (7)) and different numbers of iterations for the learning step that seeks to maximize the joint probability in Eq. (5) using the paired data $\{(\hat{z}_i, \mathbf{I}_i)\}$.

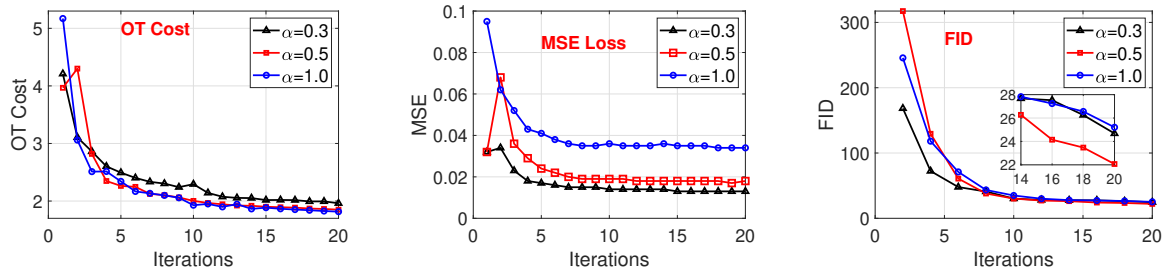


Figure 5. The influences of α on the OT cost, MSE loss and FID over different iterations.

The influence of α. Firstly, we investigate the influence of α in Eq. (10), which is shown in Fig. 5. 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.

		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

Table 4. The influence of the step size of the Langevin dynamics.

The influence of the Langevin step size. Next, we show the performances of our model with different Langevin step sizes (s in Eq. (7)) in Table 4, 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. (7) 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 5. 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.

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

Table 5. The influence of the number of Langevin steps K.

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. (5) by the paired data $\{(\hat{z}_i, \mathbf{I}_i)\}$. The results are shown in Table 6. From the table we can find that by increasing L_2, we can get much better performances for image reconstruction and generation.

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

Table 6. The influence of the number of learning iterations.

7. Conclusion

In this paper, we propose to use the OT theory to correct the bias of the short-run MCMC-based inference in training the deep latent variable models. Specifically, we correct the marginal distribution of the latent variables of the short-run Langevin dynamics through the OT map between this distribution and the prior distribution step by step. In such a way, the distribution of the inferred latent vectors will finally converge to the prior distribution, thus improving the accuracy of the subsequent parameter learning. Experimental results show that the proposed training method performs better than the ABP and VAE models on the tasks like image reconstruction, image generation and anomaly detection.

References

[1] Dongsheng An, Yang Guo, Na Lei, Zhongxuan Luo, Shing-Tung Yau, and Xianfeng Gu. AE-OT: A new generative model based on extended semi-discrete optimal transport. In *International Conference on Learning Representations (ICLR)*, 2020. 2, 5

[2] Dongsheng An, Yang Guo, Min Zhang, Xin Qi, Na Lei, and Xianfang Gu. AE-OT-GAN: Training GANs from data specific latent distribution. In *European Conference on Computer Vision (ECCV)*, page 548–564, 2020. 2

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, pages 214–223, 2017. 2

[4] Tristan Aumentado-Armstrong, Stavros Tsogkas, Allan Jepson, and Sven Dickinson. Geometric disentanglement for generative latent shape models. In *International Conference on Computer Vision (ICCV)*, pages 8180–8189, 2019. 1

[5] F. Aurenhammer, F. Hoffmann, and B. Aronov. Minkowski-type theorems and least-squares clustering. *Algorithmica*, 20(1):61–76, 1998. 5

[6] Adrian Barbu and Song-Chun Zhu. *Monte Carlo Methods*. Springer, 2020. 1

[7] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transportation distances. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2292–2300, 2013. 5

[8] Bin Dai and David Wipf. Diagnosing and enhancing VAE models. In *International Conference on Learning Representations (ICLR)*, 2019. 6

[9] Partha Ghosh, Mehdi S. M. Sajjadi, Antonio Vergari, Michael Black, and Bernhard Scholkopf. From variational to deterministic autoencoders. In *International Conference on Learning Representations (ICLR)*, 2020. 6

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014. 1, 2

[11] David Xianfeng Gu, Feng Luo, jian Sun, and Shing-Tung Yau. Variational principles for minkowski type problems, discrete optimal transport, and discrete monge-ampère equations. *Asian Journal of Mathematics*, 20(2):383–398, 2016. 5

[12] Jinjin Gu, Yujun Shen, and Bolei Zhou. Image processing using multi-code GAN prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3009–3018, 2020. 1

[13] Tian Han, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. Alternating back-propagation for generator network. In *The AAAI Conference on Artificial Intelligence (AAAI)*, pages 1976–1984, 2017. 1, 2, 3, 4, 6, 7

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a nash equilibrium. *Advances in Neural Information Processing Systems (NIPS)*, pages 6626–6637, 2017. 6

[15] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2014. 5

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2013. 1, 2, 6, 7

[17] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 1955. 5

[18] Rithesh Kumar, Anirudh Goyal, Aaron C. Courville, and Yoshua Bengio. Maximum entropy generators for energy-based models. *arXiv:1901.08508*, 2019. 7

[19] Paul Langevin. On the theory of brownian motion. *Journal of Statistical Physics*, 1908. 2, 3

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998. 5, 6

[21] Hristian Ledig, Lucas Theis, Ferenc Huszan, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 105–114, 2017. 1

[22] 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)*, pages 424–433, 2014. 5

[23] Yu-Jhe Li, Yun-Chun Chen, Yen-Yu Lin, Xiaofei Du, and Yu-Chiang Frank Wang. Recover and identify: A generative dual model for cross-resolution person re-identification. In *International Conference on Computer Vision (ICCV)*, pages 8089–8098, 2019. 1

[24] Jun S Liu. *Monte Carlo Strategies in Scientific Computing*. Springer Science & Business Media, 2008. 1

[25] Siddharth N, Brooks Paige, Jan-Willem van de Meent, Alban Desmaison, Noah Goodman, Pushmeet Kohli, Frank Wood, and Philip Torr. Learning disentangled representations with semi-supervised deep generative models. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5925–5935, 2017. 1

[26] Radford M Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11):2, 2011. 2

[27] Yu Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103:127–152, 2005. 5

[28] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011. 5, 6

[29] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. HoloGAN: Unsupervised learning of 3D representations from natural images. In *International Conference on Computer Vision (ICCV)*, pages 7587–7596, 2019. 1

[30] Erik Nijkamp, Bo Pang, Tian Han, Linqi Zhou, Song-Chun Zhu, and Ying Nian Wu. Learning multi-layer latent variable

model via variational optimization of short run MCMC for approximate inference. In *European Conference on Computer Vision (ECCV)*, pages 361–378, 2020. 1, 3, 4, 6, 7

[31] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1, 6, 7

[32] Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Found. Trends Mach. Learn.*, 11(5-6):355–607, 2019. 2

[33] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2352–2360, 2016. 1

[34] Shaogang Ren, Dingcheng Li, Zhixin Zhou, and Ping Li. Estimate the implicit likelihoods of GANs with application to anomaly detection. In *Proceedings of The Web Conference 2020 (WWW)*, pages 2287–2297, 2020. 1

[35] Nicki Skafte and Sø ren Hauberg. Explicit disentanglement of appearance and perspective in generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1016–1026, 2019. 1

[36] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[37] Cédric Villani. *Topics in Optimal Transportation*, volume 58. American Mathematical Society, 2003. 5

[38] Cédric Villani. *Optimal Transport: Old and New*, volume 338. Springer Science & Business Media, 2008. 2

[39] Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Learning dynamic generator model by alternating back-propagation through time. In *The AAAI Conference on Artificial Intelligence (AAAI)*, volume 33, pages 5498–5507, 2019. 1, 2

[40] Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Motion-based generator model: Unsupervised disentanglement of appearance, trackable and intrackable motions in dynamic patterns. In *The AAAI Conference on Artificial Intelligence (AAAI)*, pages 12442–12451, 2020. 1, 2

[41] Xianglei Xing, Ruiqi Gao, Tian Han, Song-Chun Zhu, and Ying Nian Wu. Deformable generator networks: unsupervised disentanglement of appearance and geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 2

[42] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient GAN-based anomaly detection. *arXiv: 1802.06222*, 2018. 7

[43] Jing Zhang, Jianwen Xie, and Nick Barnes. Learning noise-aware encoder-decoder from noisy labels by alternating back-propagation for saliency detection. In *European Conference on Computer Vision (ECCV)*, pages 349–366, 2020. 2

[44] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. From facial expression recognition to interpersonal relation prediction. *International Journal of Computer Vision (IJCV)*, 126(5):550–569, 2018. 5, 6

[45] Song-Chun Zhu and David Mumford. Grade: Gibbs reaction and diffusion equations. In *International Conference on Computer Vision (ICCV)*, pages 847–856, 1998. 3

[46] Yizhe Zhu, Jianwen Xie, Bingchen Liu, and Ahmed Elgammal. Learning feature-to-feature translator by alternating back-propagation for generative zero-shot learning. In *International Conference on Computer Vision (ICCV)*, pages 9844–9854, 2019. 2