Normalizing Flow with Variational Latent Representation

Hanze Dong∗†, Shizhe Diao∗‡, Weizhong Zhang†, and Tong Zhang†‡

†Department of Mathematics, HKUST
‡Department of Computer Science and Engineering, HKUST

Abstract

Normalizing flow (NF) has gained popularity over traditional maximum likelihood based methods due to its strong capability to model complex data distributions. However, the standard approach, which maps the observed data to a normal distribution, has difficulty in handling data distributions with multiple relatively isolated modes. To overcome this issue, we propose a new framework based on variational latent representation to improve the practical performance of NF. The idea is to replace the standard normal latent variable with a more general latent representation, jointly learned via Variational Bayes. For example, by taking the latent representation as a discrete sequence, our framework can learn a Transformer model that generates the latent sequence and an NF model that generates continuous data distribution conditioned on the sequence. The resulting method is significantly more powerful than the standard normalization flow approach for generating data distributions with multiple modes. Extensive experiments have shown the advantages of NF with variational latent representation.

1 Introduction

Modeling high dimensional data without supervision is a fundamental and challenging problem in statistical learning. Recently, deep generative modeling provides us possibilities to achieve this goal with a generative function between a known distribution $Z$ and observation data $X$. The main idea is to learn a function such that $G_{\theta}(Z) \approx X$. Unfortunately, most of these models, such as GANs [Goodfellow et al., 2014] and VAEs [Kingma and Welling, 2013], cannot provide explicit likelihood and fail to perform density evaluation for downstream statistical procedures due to the absence of $p_{\theta}(x)$ form. Another approach is the flow-based method, which has explicit likelihood and captures isomorphism between $Z$ and $X$. In other words, $G_{\theta}$ in NF is an invertible function such that $G_{\theta}^{-1}(X) \approx Z$. Recent progress in normalizing flows (NFs) [Dinh et al., 2014, Kingma and Dhariwal, 2018, Dinh et al., 2017, Durkan et al., 2019] and Neural ODEs [Chen et al., 2018, Grathwohl et al., 2018] have shown that neural networks are capable of reproducing invertible dynamical systems. Notably, when $G_{\theta}$ is invertible, the likelihood $p_{\theta}(x)$

*Equal contributions.
†Our code is available at https://github.com/hendrydong/normalizing-variational-flow.
can be explicitly expressed with the Jacobian function of $G_\theta$. Thus, NF provides us a free-form likelihood parametric modeling framework. It is quite natural and elegant to perform maximum likelihood estimation (MLE) to learn the above isomorphism.

Intuitively, due to the invertibility, we can regard $G_\theta$ as a transportation function, which conveys $Z$ to $X$. For simple transportation, such as shift and scale, $G_\theta$ can be easily learned. Thus, NF framework works well for these examples. Nonetheless, for real data, distributions are often scattered into different clusters (i.e., modes). When these clusters are isolated, the optimal $G$ would be complex, i.e., there are some “discontinuity” in the function with bad Jacobian behavior. This makes the $G_\theta$ extremely hard to learn. We can consider an example in Figure 1 that $X$ is from a Gaussian mixture model. Obviously, the continuity of transportation function is rather poor near the center of $Z$, and Gaussian initialization puts most density on this area. Thus, learning the invertible dynamics of transportation becomes highly unstable in these discontinuous part. In other words, it is rather difficult to model a scattered distribution by a function from Gaussian. Details are given in Section 2.

In this paper, to resolve the above limitation of NF, we propose a new framework called Normalizing Variational Flow, (NVF). Specifically, we decompose $X$ into parts, each of which contains only a unimodal, highly concentrated distribution cluster. From this view, we do NOT need to match $Z$ to the full space of $X$, but only find the bijection from $Z$ to each cluster of $X$ separately. Inspired by clustering techniques, we can incorporate a latent variable to denote the representation of each cluster of $X$ and simplify the transportation function with the latent knowledge. The feasibility of the idea is guaranteed by Universal Approximation Theorem of mixture model: even a mixture of Gaussian can approximate any smooth density functions [McLachlan and Peel, 2000]. For example, in Figure 1, we divide data samples into two categories 0 and 1. By introducing the latent factor $u$, the transportation function becomes linear without any discontinuity. In real applications, when such latent variables, such as image labels, attributes, are given, previous studies...
[Salimans et al., 2016, Karras et al., 2020, Grathwohl et al., 2019] have demonstrated that leveraging these variables can effectively improve the generation quality by simplifying $G_\theta$. Beyond supervised labels, for real data, the latent variables exist in a more general form: discrete, continuous, or even sequential latent representations are reasonable candidates and they are usually not accessible. Therefore, how to design a proper latent variable $u$ and how to effectively learn it become two crucial problems.

We argue that we should adopt latent variables which are more fine-grained than supervised labels. The reason is that the number of modes can be far more than that of labels. Without a proper structure of $u$, the desired transportation function cannot be obtained smoothly. In general, $u$ should contain global information of each $x$ that makes NF transportation do not need to match the whole population, but one mode of it. For different data structures and modeling purposes, the design and parameterization of $u$ varies: For tabular data or toy data, the key latent variable is the cluster center, we use finite categories as the discrete probability space of $u$; For complex data, such as images and texts, the cluster center cannot be easily expressed as a single vector, so we introduce a general sequential representation for them. [Dosovitskiy et al., 2020] stress that images should be represented as $16 \times 16$ sequences.

Our approach supports multiple forms of $u$ that are preferred in different settings, which are important in real practice. Specifically, we adopt and extend Variational Bayesian (VB) method to jointly learn $G_\theta$ and the latent variable $u$. The posterior $\pi_\theta(u|x)$ is our target to assign $u$ for each $x$. Existing MLE based methods, such as expectation–maximization (EM) algorithm [Moon, 1996], necessitate the explicit form probability measure of $\pi_\theta(u|x)$ in the parametric model. However, this strategy is intractable due to the traversal requirement of all possible $u$. Variational Bayesian methods provide us an alternative to learn another probability $\nu_\phi(u|x)$ to approximate $\pi_\theta(u|x)$. In our context, this $\nu_\phi(u|x)$ can be parameterized by a neural network encoder, which is flexible about the type of $u$. By incorporating Variational Bayes, the NF framework can be optimized with evidence lower bound (ELBO). This ELBO-based framework mitigates the fatal issue caused by ill-behaved Jacobian matrix in NF.

**Contributions.**

- We point out that conventional NF that maps the observed data to a normal distribution has difficulty in handling data distributions with multiple isolated modes;
- We propose a framework to optimize NF models based on VB that resolves the aforementioned issue. It employs a variational latent representation to model cluster information, so that a simplified transportation function can be used for each cluster;
- Our variational latent framework supports different types of latent factor $u$ that can be customized for different data modeling tasks;
- Experiments demonstrated the importance of variational latent representation for NF, and the effectiveness of NVF.
2 Background

Notations. Throughout this paper, capital letters denote random variables and lower-case letters denote data samples. $p_\theta$ denotes the density parameterized by $\theta$, $p_X$ and $p_Z$ denote data distribution $X$ and a known density of $Z$, respectively. $\nabla G_\theta(x)$ and $\nabla_x G_\theta(x)$ denote the Jacobian of $G$ w.r.t. $x$. $\nabla_\theta G_\theta(x)$ is the Jacobian w.r.t. $\theta$. $|J|$ is the determinant of a matrix $J$. $X$ and $Z$ are continuous random variables. For latent random variable $U$, it can be continuous, discrete, or sequential. The probability measure induced by $\beta$ and $\phi$ is $\pi_\beta$ and $\nu_\phi$, the Radon–Nikodym derivative [Durrett, 2019] of $\pi_\beta$ and $\nu_\phi$ over the measure $\mu_u$ of $U$ is denoted by $p_\beta$ and $p_\phi$, which is the generalized definition for both discrete and continuous random variables. $\text{KL}(p||q)$ denotes the Kullback–Leibler (KL) divergence of $p$ from $q$. $\mathcal{N}(\mu, \Sigma)$ denotes the Gaussian distribution with mean/covariance $\mu$ and $\Sigma$.

Normalizing Flows and Latent Variable. This work considers the dynamics of normalizing flow, which aims to find an invertible transportation between two random variables. Given data distribution $X$, our target is to find a bijection to transform some known distribution $Z$ to $X$, i.e., $G(Z) = X$ and $Z = G^{-1}(X)$. The equalities hold only if $X$ and $Z$ have same dimensions and the Jacobian of $G$ is positive definite almost everywhere. Then we can use a change of variable to get the density of $X$: $p_X(x) = p_Z(G^{-1}(x)) |\nabla G^{-1}(x)|$. This strategy naturally leads to the MLE optimization of NF. Consider a parameterized model $G_\theta$, the log-likelihood can be written as

$$\log p_\theta(x) = \log p_Z(G_\theta^{-1}(x)) + \log |\nabla G_\theta^{-1}(x)|. \quad (2.1)$$

Thus, MLE of $\theta$ can be obtained with $\nabla G$ and $G_\theta^{-1}$. This method is a free-form likelihood modeling: When $G_\theta$ is capable of transporting $Z$ to $X$, it can be successfully modelled without
Figure 3: Clustering phenomenon brings ill-behaved Jacobian

a specific standard distribution class. When both X and Z are continuous, there exists an isomorphism $G_*$ between Z and X (both of them are standard probability space [Rokhlin, 1949]), i.e., $G_*^{-1}(X) = Z$, $G_*(Z) = X$. The only remaining issue is to approximate $G_*$ with a parameterized model $G_\theta$. When $G_*$ is well-behaved, the approximation procedure can be done smoothly. Nonetheless, this strategy fails when X is scattered into several clusters. Our next example verifies this phenomenon.

Example 1. Consider that X is a 1-d Gaussian mixture model with two components $\mathcal{N}(-\mu, \sigma^2)$, $\mathcal{N}(\mu, \sigma^2)$. We assume that $\mu > \sigma > 0$ and $G_*$ is the optimal transportation function from Z to X: $G_*(Z) = X$ and $G_*^{-1}(X) = Z$. Assume that (1) Z is $\mathcal{N}(0, 1)$, then $|\nabla G_*(0)| = \sigma \exp\left(\frac{\mu^2}{2\sigma^2}\right)$. (2) Z is a mixture of $\mathcal{N}(-\mu/2, \sigma^2)$ and $\mathcal{N}(\mu/2, \sigma^2)$, then $|\nabla G_*(0)| = \exp\left(\frac{3\mu^2}{8\sigma^2}\right)$.

Example 2. If we have latent variable u to denote the cluster information of Example 1 and we use $Z = \mathcal{N}(0, 1)$. Then for $u = 0$, $G_*(z, u = 0) = \sigma z - \mu$; for $u = 1$, $G_*(z, u = 1) = \sigma z + \mu$. Both functions are linear and the Jacobian is a small constant, i.e., $\nabla G_* = \sigma$.

Example 1 indicates that the Jacobian at zero can be exponentially large w.r.t. $\mu/\sigma$. We have some examples with different $\mu$ and $\sigma$ in the Appendix. For those scattered datasets, this property is fatal. Our case (2) shows that even we use a multi-clustered Z in the example, without proper alignment with cluster center $\mu$, the phenomenon cannot be resolved. If we directly restrict $G_\theta$ to have bounded $|\nabla G_\theta|$ or $|\nabla^2 G_\theta|$, then the approximated transportation function is far away from $G_*$, which is empirically verified by the examples in Figure 2. It is shown that with the restricted Jacobian or Hessian matrix, the transportation function from Z to X cannot be achieved with...
MLE and the resulting density estimation is very poor. Hence, the idea of normalizing flow based on the plain transportation function is not reasonable in this case. The next example sheds light on the construction of proper transportation function to avoid this phenomenon. We visualize the ill-behaved transportation functions with different $\mu$ and $\sigma$ in Figure 3.

Example 2 shows that the discontinuity issue can be eliminated by introducing the latent variable $u$ into the model. We will follow this motivation to build up our algorithm.

3 Normalizing Flow with Variational Bayes

In this part, we do NOT treat NF as a black box which finds MLE without further consideration about data distribution shape, which can be captured by latent representation $u$. In contrast, we introduce the latent representation parameterized by $\phi$ to simplify $G_\theta$ and develop an algorithm based on NF to joint learn the transportation function and latent representation.

Intuitively, we first investigate data distribution properties to choose a proper type of latent representations, and then use it to simplify the learning process of the transportation function. Similar to Figure 1, the introduction of latent representations enables us to learn the transportation function for different modes separately.

3.1 Design of Latent Representation

Since $u$ represents the properties of a data sample $x$, for each $x$, we aims to find $u_x$ to represent the cluster (mode) information of $x$. A proper space $U$ of the latent representation matters in the simplification of the transportation function. We argue that the $U$ space should be capable to summarize the full cluster/mode information of data distributions. Hence, it is usually more fine-grained than supervised labels. In the following parts, we will build up the construction and learning process of $U$ space.

Inspired by existing practical applications, we model the latent space as the following categories [Kingma and Welling, 2013, Oord et al., 2017, Dosovitskiy et al., 2020]: (1) DISCRETE space: when the data are low-dimension with limited modes, such as tabular data, the discrete space is sufficient for cluster information of $X$; (2) CONTINUOUS space: when the space of $x$ is close to a smooth manifold, continuous latent $u$ is a natural choice. For example, the geodesic distance from starting point is a proper $u$ for Swiss roll data. (3) SEQUENTIAL space\(^2\): for images, $16 \times 16$ sequence is needed as a proper representation [Dosovitskiy et al., 2020]. Modern text representations are also based on sequences [Devlin et al., 2019, Liu et al., 2019, Radford et al., 2019, Brown et al., 2020]. More detailed descriptions of the properties are included in Appendix.

Specifically, we model $u$ as a random variable with prior $\pi_\beta(u)$, parameterized by $\beta$, which is required in our NF model. Thus, the posterior $\pi_\theta(u|x) = p_\theta(x|u)\pi_\beta(u)/p_\theta(x)$ also be well-defined\(^3\).

\(^2\)In a broad sense, the sequential latent space is also a discrete space, but the number of state of a sequence is usually too large to be traversed, thus we consider it as a new case.

\(^3\)Strictly speaking, the $\pi_\theta(u|x)$ should be written as $\pi_{\theta,\beta}(u|x)$. We omit the $\beta$ in notation for simplicity without ambiguity.
Table 1: Generative modeling comparison. (∗: With discrete latent, NVF obtains exact $p_\theta$)

| Model                                      | GAN | Neural-SDE | Neural-ODE | AR | VAE | NF | NVF |
|---------------------------------------------|-----|------------|------------|----|-----|----|-----|
| Free-form likelihood                        | -   | ✓          | ✓          |    |     | ✓  | ✓   |
| Well-behaved $|\nabla G|$ | -   | ✓          | ✓          | ✓  | ✓   | ✓  | ✓   |
| Exact density estimation                    |    |            | ✓          |    | ✓   | ✓  | ✓   |
| Density approximation                       | ✓   | ✓          | ✓          | ✓  | ✓   | ✓  | ✓   |
| Explicit generator                          | ✓   | ✓          | ✓          | ✓  | ✓   | ✓  | ✓   |
| Continuous data sampling                    | ✓   | ✓          | ✓          | ✓  | ✓   | ✓  | ✓   |
| Explicit $u|x$ or $z|x$                      | ✓   | ✓          | ✓          | ✓  | ✓   | ✓  | ✓   |

Figure 4: NF with variational latent representation: sample $x$ is generated by a latent representation $u$ and noise $z$. Solid lines denote the NF model $G_\theta$ with latent prior $p_\beta$, such that $p_{\theta,\beta}(x, u, z) = p_\beta(u)p_Z(z)p_\theta(x|u, z)$, where $p_\theta(x|u, z) = 1$ if $x = G_\theta(z, u)$, $p_\theta(x|u, z) = 0$ if $x \neq G_\theta(z, u)$ which can also be reformulated as $p_\beta(u)p_Z(G_\theta^{-1}(x, u)) |\nabla_x G_\theta^{-1}(x, u)| p_\theta(z|x, u)$. Dotted line denotes variational approximation $\nu_\phi(u|x)$ of intractable posterior $\pi_\theta(u|x)$. Dashed lines denote the tractable posterior $\pi_\theta(z|x, u)$.

Then, the learning framework can be established.

### 3.2 Variational Latent Posterior

We turn to present our framework to optimize $G_\theta$ and latent representation jointly. Given latent $u$, and the (prior) probability measure $\pi_\beta(u)$, we have

$$p_\theta(x|u) = p_Z(G_\theta^{-1}(x, u)) |\nabla_x G_\theta^{-1}(x, u)|,$$  

(3.1)

$$p_\theta(x) = \int p_\theta(x|u)d\pi_\beta$$

$$= \int p_Z(G_\theta^{-1}(x, u)) |\nabla_x G_\theta^{-1}(x, u)| d\pi_\beta. $$  

(3.2)

where $\pi_\beta(u)$ is the probability measure defined by $\beta$ that can be either continuous or discrete.
Conventional NF methods would prefer to optimize Eq. (3.2) analytically, which is not sensible in our case: (1) For small discrete space, although it is possible to compute the exact loss, the introduction of \( u \) space makes the density evaluation entail the traversal of \( U \) space, while most \( \pi_\theta(u|x) \) are near 0 due to the clustering phenomenon; (2) When the space is large, such as continuous and sequential space, the exact optimization of Eq. (3.2) becomes intractable. Intuitively, for a fixed \( x \), it is wasteful to traverse the \( U \) space, since only a few of \( u \) are important. Thus, we resort to a more economical way to approximate \( p_\theta(x) \): we aim to obtain \( u|x \) efficiently and only sample important ones.

Variational inference provides us the possibility to find the posterior. In particular, Stochastic Gradient Variational Bayes estimator enables us to obtain a variational approximator \( \nu_\phi(u|x) \) of intractable posterior \( \pi_\theta(u|x) \). Rather than optimizing log-likelihood, we consider to optimize evidence lower bound (ELBO) to obtain \( G_\theta \) and \( \nu_\phi(u|x) \). The full diagram of our proposed framework is displayed in Figure 4.

By Jensen’s inequality, Proposition 1 replaces the Eq. (3.2) with a lower bound, leading to our variational formulation.

**Proposition 1.** In our setting, the evidence lower bound (ELBO) of log-likelihood is written as

\[
\int \log p_\theta(x|u) d\nu_\phi(u|x) - KL(\nu_\phi(u|x)\|\pi_\beta(u)).
\]

(3.3)

For efficient computation, we use Monte Carlo samples to estimate ELBO,

\[
\frac{1}{K} \sum_{k=1}^{K} \log p_\theta(x|u_{xk}) - KL(\nu_\phi(u|x)\|\pi_\beta(u)),
\]

(3.4)

where \( u_{xk} \) are i.i.d. samples from \( \nu_\phi(u|x) \).

By taking Eq. (3.1) and (3.4) together, we can get the loss function below.

**Proposition 2.** For each data sample \( x \), we can obtain our loss function

\[
\ell(\theta, \phi, \beta; x) = -\frac{1}{K} \sum_{k=1}^{K} \left[ \log p_Z(G_{\theta}^{-1}(x, u_{xk})) + \log |\nabla_x G_{\theta}^{-1}(x, u)| \right]
+ KL(\nu_\phi(u|x)\|\pi_\beta(u))
\]

(3.5)

where \( u_{xk} \) are samples from \( \nu_\phi(u|x) \).

We consider two prior parameterization strategies for different latent space (1) For discrete or continuous space, the prior \( \pi_\beta(u) \) is given by uninformative distributions such as uniform distribution, Gaussian distribution, which means \( \beta \) is not need to be learned; (2) For sequential space, the learning process the \( \pi_\beta \) is important. With the a powerful \( \pi_\beta \) class, there exists \( \pi_\beta(u) \) such that \( \pi_\beta(u) = E_X \nu_\phi(u|x) \), and the KL term can be eliminated. In such the case, the loss function can be reduced (Proposition 3).
Proposition 3. If \( \beta \) is learnable and \( \nu_\phi \) is in the distribution class of \( \pi_\beta \). Eq. (3.5) can be equivalently rewritten as a two separate loss functions for \( \phi \), \( \theta \), and \( \beta \).

1. Loss function of \( \theta \) and \( \phi \)
\[
\ell_1(\theta, \phi; x) = -\frac{1}{K} \sum_{k=1}^{K} \log p_Z(G^{-1}_\theta(x, u_{xk})) - \log |\nabla G^{-1}_\theta(x)|; 
\]

2. Given \( \phi \), the loss function of \( \beta \) is
\[
\ell_2(\beta; x, \phi) = -\frac{1}{K} \sum_{i=1}^{K} \log \frac{d\pi_\beta(u_i)}{d\mu_u}, \ u_i \sim \nu_\phi(u|x), 
\]
where we take the Radon–Nikodym derivative of \( \pi_\beta \) over the measure \( \mu_u \) of \( U \).

Thus, when \( \beta \) is fixed, we minimize Eq. (3.5) to obtain \( \hat{\theta}, \hat{\phi} \), when \( \beta \) is learnable, minimization of Eq. (3.6) and (3.7) leads to the estimator of \( \hat{\theta}, \hat{\phi}, \hat{\beta} \).

3.3 NVF Algorithm

To be more clear, we introduce an encoder \( E_\phi(x) \) as the neural network realization of \( \nu_\phi(u|x) \). For discrete \( u \), the output of \( E_\phi(x) \) is the probability of each state \( v_i \):
\[
E_\phi(x) = [\nu_\phi(v_1|x), \cdots, \nu_\phi(v_s|x)]^T.
\]
Sampling from \( \nu_\phi(u|x) \) can be done by Gumble-Softmax technique [Jang et al., 2016]. For continuous random variable, we constrain the posterior \( \nu_\phi(u|x) \) can only be Gaussian, which makes output of \( E_\phi(x) \) as \( \mu \) and \( \sigma \) for the variational posterior \( N(\mu, \sigma I) \). The sampling process is done by reparameterization trick [Kingma and Welling, 2013]. For sequential \( U \) space, \( u_x = E_\phi(x) \) is a latent sequence representation, \( \log \nu_\phi(u = u_x|x) \propto -||u - u_x||^2 \), \( u_x \) is learned by vector quantization [Oord et al., 2017]. \( \nu_\phi(u|x) \) sample the most likely \( u \). The specification of posterior sampling enable us to present full algorithm.

For the evaluation stage, if \( u \) is discrete, we can use Eq. (3.2) to evaluate density. For continuous latent space, some approximation methods are proposed [Burda et al., 2015], but \( p_\theta(x) \) cannot be obtained efficiently, we will evaluate \( p_\theta \) for this case. If \( u \) is sequential, we can use \( E_\phi(x) \) to obtain the most likely \( v_{x1}, \cdots, v_{xK} \in U \) that are top-\( K \) \( \nu_\phi(u|x) \), and compute \( \hat{p}_\theta(x) = \sum_{i=1}^{K} \pi_\theta(u = v_{xi})p(x|v_{xi}) \).

3.4 NVF v.s. Other Generative Models

In this part, we compare our NVF with other generative models. Although GAN is popular in applications [Goodfellow et al., 2014, Brock et al., 2018], the statistical properties are missing in GAN’s framework. The likelihood of is missing and we cannot even approximate the density of data samples. GAN-based models are based on discriminator network to approximate likelihood in feature space, whose statistical properties are vague [Goodfellow et al., 2014, Dinh et al., 2017]. Transportation based methods, such as Neural ODE/SDE and NF, suffer from the Jacobian issue.
Algorithm 1 NVF: Normalizing Variational Flow

**Input:** Data observations \( D = \{x_1, \cdots, x_n\} \); Parameter initialization \( \theta, \phi, \) and \( \beta \). Learning rate \( \eta \). Batch size \( b \).

**Output:** The learned NF \( G_\theta \); variational latent posterior approximator \( \nu_\phi \); the learned prior \( \pi_\beta \) if it is learnable.

1: while \( \theta \) and \( \phi \) do not converge do
2: Sample data \( x_{(i)} \in D \), for \( i = 1, \cdots, b \).
   For each \( x_{(i)} \), compute \( \nu_\phi(u|x_{(i)}) \) and sample \( u_{x_{(i)}}k \) from \( \nu_\phi(u|x_{(i)}) \), for \( k = 1, \cdots, K \).
3: For each \( u_{x_{(i)}}k \), obtain the map on \( Z \) space such that \( z_{x_{(i)}}k = G_\theta^{-1}(x, u_{x_{(i)}}k) \).
4: if \( \beta \) is not learnable then
5: \( [\theta, \phi] \leftarrow [\theta, \phi] - \eta b^{-1} \sum_{i=1}^b [\partial \ell(\theta, \phi, \beta; x_{(i)}) \partial \theta, \partial \ell(\theta, \phi, \beta; x_{(i)}) \partial \phi] \) in Eq. (3.5).
6: else
7: \( [\theta, \phi] \leftarrow [\theta, \phi] - \eta b^{-1} \sum_{i=1}^b [\partial \ell_1(\theta, \phi; x_{(i)}) \partial \theta, \partial \ell_1(\theta, \phi; x_{(i)}) \partial \phi] \) in Eq. (3.6).
8: \( \beta \leftarrow \beta - \eta b^{-1} \sum_{i=1}^b \frac{\partial \ell_2(\beta; x_{(i)}, \phi)}{\partial \beta} \) in Eq. (3.7).
9: end if
10: end while

as described in Section 2. In particular, Neural-ODE/SDE utilizes an integral to obtain the transportation rather than an explicit function, which entails extensive computation. The introduction of noise in Neural-SDE makes it unable to find the latent \( z \) or \( u \) reversely. Autoregressive (AR) models are based on discrete/sequence modeling, which does not support continuous data sampling. Compared with VAE, NVF integrates the NF framework into the likelihood modeling, which supports free-form density function and more complex data modeling, such as images. Besides, VAE is based on the Gaussian likelihood assumption, whose modeling capacity is highly restricted. Our NVF supports exact density estimation that cannot be accomplished by conventional VB frameworks. Table 1 summarizes the comparison.

In summary, our framework shows privileges in the following scenario: (1) Isolated latent cluster representations: Data distribution \( X \) is constructed with both cluster center and noise nearby. The cluster representation usually captures global information of \( X \), denoted as \( U \). The noise part is the quotient space of \( X \) by \( U \), i.e., the space obtained by “collapsing” \( U \) component to zero. We focus on the case that the elements of \( U \) are isolated. The \( \nabla G \) of conventional transportation based methods (NF, Neural-ODE/SDE) would be ill-behaved. (2) Non-Gaussian noise space: the noise near each cluster is a complex distribution rather than Gaussian noise. The Gaussian likelihood assumption oversimplify the real practice. (3) Limited \( G_\theta \) capacity: The target \( G_\theta \) with ill-behaved Jacobian/Hessian matrix cannot be achieved. The ill-behaved \( \nabla G \) fails modeling.

These three conditions are automatically satisfied for most real data and parameterized function classes. We take visual datasets for example: (1) the visual datasets usually contain multiple labels and attributes. Some samples sharing the same properties belong to the same cluster. (2) The noise in image datasets varies, such as background noise, compression noise, which are not Gaussian; (3) For any parameterized function class, the capacity is limited for model size and numerical stability consideration. The non-smoothness and non-continuity should be avoid.
4 Related Works

4.1 Normalizing Flow

Normalizing flow (NF) is a popular and powerful framework in generative modeling. The core idea of NF is to construct a reversible generative function $G_\theta$ between a known distribution $Z$ and observation $X$, and use the change of variables formula to model the distribution of $X$. Thus the density of $X$ can be computed by $p_Z(G^{-1}(x)) |\nabla G^{-1}(x)|$. The key challenge for this framework is to obtain the inverse form of $G_\theta$ and to compute Jacobian determinant. Thus, early works [Dinh et al., 2014, Kingma and Dhariwal, 2018, Dinh et al., 2017, Durkan et al., 2019] target at constructing special functions of $G_\theta$ that has explicit inverse and simple Jacobian form. Some ODE-based methods [Chen et al., 2018, Grathwohl et al., 2018] also consider to compute an integral to make the Jacobian more flexible. These algorithms make it possible to compute the exact likelihood of data samples. Due to the explicit form of $p_\theta(x)$ for input $x$, we can perform density estimation in NF framework. Besides, the sampling procedure can also be done painlessly with the obtained $G_\theta(z)$.

Nonetheless, the transportation function might be complex. Conventional normalizing flow models [Dinh et al., 2014, Dinh et al., 2017, Kingma and Dhariwal, 2018, Durkan et al., 2019] ignore this challenge. Interestingly, the label information of the datasets, which helps to construct better generation, can also be recognized as a latent variable [Brock et al., 2018]. Our motivation originates from these “labels” and we extend them to general unsupervised latent variables that help transportation.

4.2 Variational Bayes

Variational Bayes (VB) is a method to approximate the intractable posterior $\pi_\theta(u|x)$ of latent variables, which cannot be solved by conventional statistical methods such as EM algorithm. The main idea of VB is to use another parametric model $\nu_\phi(u|x)$ to approximate $\pi_\theta(u|x)$. Thus, the parameterization of $\phi$ brings approximation error regarding the capacity. Variational auto-encoder (VAE) [Kingma and Welling, 2013] is the most popular example that uses encoder-decoder neural network pair as the parametric model. However, the distributions of both $\pi_\theta(x|u)$ and $\nu_\phi(u|x)$ are only restricted to standard distribution classes, such as diagonal Gaussian distribution. VQ-VAE [Oord et al., 2017, Razavi et al., 2019] is an extension of VAE that leverages sequential latent variable. The distribution class of $\pi_\theta(x|u)$ in VQ-VAE is still restricted. It only considers $\pi_\theta(x|u)$ as a Gaussian distribution.

Since NF has the capacity of modeling complex densities, it is suitable for the variational approximation step [Rezende and Mohamed, 2015]. Previous work use NF for posterior approximation (VI with NF). In other words, NF was only used to model $\nu_\phi(u|x)$. On the contrary, our framework (NF with VI) does not concentrate on complex posterior modeling but enhances NF modeling with another latent variable. It is also reasonable to consider this framework as an extension of VB that uses NF to improve the capacity of $\pi_\theta(x|u)$. Note that when $u$ is finite, our framework provide explicit likelihood form, which cannot be obtained by VAE formulation. We highlight that even finite latent $u$ still help NF significantly.
Figure 5: Comparison between NF and our NVF. Size indicates the flow depth × flow width. Param. indicates the parameter size. For NF, we use standard Gaussian as $Z$. For NVF, the posterior approximator $\phi$ is with parameter 0.9K.

5 Experiments

We implement the variational latent encoder jointly with conditional normalizing flow. To justify the significance of our NVF framework, we evaluate the likelihood of our model on several datasets, such as 2d toy data, tabular data, and visual data. To provide more intuitive explanation and illustration, we provide visualization and generation results on toy and visual data, which can help readers to understand how NVF works.

5.1 Toy Data Illustration

First, we explore the representative power of our model on toy datasets, i.e., 2-d continuous data distributions. The data are generated with `sklearn`. The sample size of training data is 1024. Figure 5 shows the comparison between NF and NVF. Several observations are made. First, with the increase of flow depths, flow widths and parameters, all three models are improved substantially and thus make a better approximation for $G_s$. Second, for NF models, it could not separate the modes, which is consistent with Example 1. Last, our model achieves a significant improvement on fitting the moon shape and Gaussian mixture, and the modes in NVF are separated clearly. These observations demonstrate that the endowing the normalizing flow with variational latent representation provides more powerful generative modeling capability and thus is able to fit the density better.
Table 2: Test negative log-likelihood for tabular dataset density estimation benchmarks (in nats); lower is better. (P: POWER; G: GAS; H: HEPMASS; M: MINIBOONE; B: BSDS300) The upper half denotes coupling-based flow networks; the lower half inverse/mask-autoregressive-based ones.

| Dataset   | P    | G    | H    | M    | B    |
|-----------|------|------|------|------|------|
| FFJORD    | -0.46| -8.59| 14.11| 10.43| -157.40 |
| RealNVP   | -0.17| -8.33| 18.71| 13.55| -153.28 |
| Glow      | -0.17| -8.15| 18.92| 11.35| -155.07 |
| NSF (C)   | -0.64| -13.09| 14.75| 9.67| -157.54 |
| Ours      | **-0.68**| **-13.21**| **14.53**| **7.35**| **-158.25** |
| MADE      | 3.08 | -3.56| 20.98| 15.59| -148.85 |
| MAF       | -0.24| -10.08| 17.70| 11.75| -155.69 |
| TAN       | -0.48| -11.19| 15.12| 11.01| -157.03 |
| NAF       | -0.62| -11.96| 15.09| 8.86| -157.73 |
| SOS       | -0.60| -11.99| 15.15| 8.90| -157.48 |
| NSF (A)   | -0.66| -13.09| 14.01| 9.22| -157.31 |
| Ours      | **-0.69**| **-13.27**| **13.87**| **7.42**| **-158.26** |

5.2 Latent space and likelihood design

In addition, we conduct experiments on CelebA [Liu et al., 2015] to discuss the effects of latent space and likelihood design from both quantitative (Table 4) and qualitative results (Fig. 6). The hyper-parameter settings follow the previous section and six models are considered: VAE, VAE with discrete latent space, VQ-VAE, normalizing flow with continuous latent, normalizing flow with discrete latent and normalizing flow with sequence latent (i.e., NVF). First, as for the likelihood design, we observe that NF-based models outperforms VAE-based models, demonstrating the advantages of the free-form likelihood. The generation samples of VAE-based method can be recognized as “cluster centers” in our literature, since it assume the Gaussian likelihood in the model, and the nearby noise is isotropic. Our NVF does not add too much restrictions on likelihood design, leading to better modeling. Furthermore, as for the latent space design, the sequence latent space performs much better than continuous and discrete latent space. Overall, normalizing flow with continuous latent performs the second best, verifying the good capability of normalizing flow and NVF achieves further improvement upon it, illustrating the power of sequence latent space. We also include the generated samples in the following sections for further comparison.
Table 3: Test negative log-likelihood for visual datasets (in bits/dim); lower is better.

| Dataset       | MNIST | Fashion | CIFAR10 (5-bit) | CIFAR10 (8-bit) | IMAGENET |
|---------------|-------|---------|-----------------|-----------------|----------|
| REALNVP       | 1.06  | 2.85   | -               | 3.49            | 3.98     |
| GLOW          | 1.05  | 2.95   | -               | 3.35            | 3.81     |
| NSF           | -     | -      | 1.67            | 3.38            | 3.82     |
| MARprior      | 0.88  | -      | -               | 3.24            | 3.80     |
| RESIDUALFLOW  | 0.97  | -      | -               | 3.08            | 3.75     |
| FLOW++        | -     | -      | -               | 3.08            | 3.69     |
| SURVAE        | -     | -      | -               | 3.08            | 3.70     |
| **Ours**      | **0.78** | **2.02** | **1.37**       | **2.98**        | **3.49** |

Table 4: FID scores on CelebA; lower is better. Note that we display the reconstructed images for VQ-VAE due to the unsatisfied quality of generation. We compare continuous latent, discrete latent and sequence latent, separately.

| Latent | Continuous | Discrete | Sequential |
|--------|------------|----------|------------|
| FID (vae) | 47.82     | 126.27   | 31.57      |
| FID (ours) | **18.85** | **68.34** | **16.23** |

5.3 Density estimation

We also conduct density estimation experiments on four UCI datasets with different dimensions $d$ [Asuncion and Newman, 2007], namely POWER ($d = 6$), GAS ($d = 8$), HEPMASS ($d = 21$), MINIBOONE ($d = 43$); and BSDS300 dataset ($d = 63$) [Martin et al., 2001] with the same data pre-processing steps following [Papamakarios et al., 2017]. The neural network architecture of $G_{\theta}$ follows NSF [Durkan et al., 2019], with 10 or 20 flow depths depending on the data. The variational posterior approximator is a 4-layer MLP encoder with 256 neurons and ReLU activation function. The number of discrete latent states is cross-validated chosen from $\{2, 5, 10, 20\}$. NF has two structures in density estimation: For both coupling based [Dinh et al., 2017, Kingma and Dhariwal, 2018, Durkan et al., 2019] and inverse/mask autoregressive based [Papamakarios et al., 2017, Germain et al., 2015, Kingma et al., 2016, Jaini et al., 2019, Oliva et al., 2018, Huang et al., 2018] approaches, the introduction of latent representation helps significantly. The results are shown in Table 2. First, our algorithm achieves state-of-the-art performance in all density estimation benchmarks. Moreover, even for FFJORD which is an ODE-based algorithm with much larger capacity, our algorithm still outperform it consistently. Last, compared with our base architecture model NSF, our improvement is significant, especially for those datasets with higher dimensions.
5.4 Normalizing Flows on Visual Datasets

In this section, we investigate the effectiveness of our algorithm on five real-world visual datasets: MNIST [LeCun, 1998], Fashion MNIST [Xiao et al., 2017], CIFAR-10 [Krizhevsky et al., 2009], CelebA [Liu et al., 2015], ImageNet [Deng et al., 2009]. Our model is trained with learning rate $10^{-4}$. We apply the Adam optimizer [Kingma and Ba, 2015] and anneal the learning rate according to a cosine schedule. We compare our model with Neural Spline Flow (NSF) [Durkan et al., 2019], Glow [Kingma and Dhariwal, 2018], MARPrior [Mahajan et al., 2020], Flow++ [Ho et al., 2019], ResidualFlow [Chen et al., 2019], in terms of negative log-likelihood on the test set. The results are shown in Table 3. It is observed that our model, NVF, achieves the state-of-the-art on all datasets and there are two major performance improvement. There is a significant jump in performance occurs between the baseline models and our NVF. This increase can be attributed to the power of the variational latent representation. Particularly, it is observed that our model achieves the best performance against all baseline models stuck on these datasets.

We provide additional samples to show the importance of latent capacity and the effectiveness of NF framework in Figure 7. It is clear that discrete and standard VAE is blurred, which means the capacity of Gaussian likelihood is insufficient to represent images. VQ-VAE can improve the performance, but the performance is still unsatisfactory (lack of details). For NF variants, due to the capacity of NF framework against Gaussian, the details can be filled. Besides, our NVF framework is the best choice to regenerate proper samples.
6 Conclusion

This paper introduces new framework for learning normalizing flow models, which incorporate a variational latent representation to simplify the transportation function. We have shown that without the latent representation, NF fails to model data with multiple isolated modes. Our variational latent representation endows our model with strong compatibility that supports different data structures, such as images. Empirical results have shown the necessity of latent variable and the significant improvement of our framework.

Acknowledgments

The work was supported by the General Research Fund (GRF 16310222 and GRF 16201320).
References

[Asuncion and Newman, 2007] Asuncion, A. and Newman, D. (2007). UCI Machine Learning Repository.

[Brock et al., 2018] Brock, A., Donahue, J., and Simonyan, K. (2018). Large Scale GAN Training for High Fidelity Natural Image Synthesis. arXiv preprint arXiv:1809.11096.

[Brown et al., 2020] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. In Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc.

[Burda et al., 2015] Burda, Y., Grosse, R., and Salakhutdinov, R. (2015). Importance Weighted Autoencoders. arXiv preprint arXiv:1509.00519.

[Chen et al., 2019] Chen, R. T., Behrmann, J., Duvenaud, D. K., and Jacobsen, J.-H. (2019). Residual flows for invertible generative modeling. Advances in Neural Information Processing Systems, 32.

[Chen et al., 2018] Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. (2018). Neural Ordinary Differential Equations. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, pages 6572–6583.

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In NAACL-HLT (1).

[Dinh et al., 2014] Dinh, L., Krueger, D., and Bengio, Y. (2014). Nice: Non-linear Independent Components Estimation. arXiv preprint arXiv:1410.8516.

[Dinh et al., 2017] Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2017). Density Estimation using Real NVP. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net.

[Dosovitskiy et al., 2020] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv preprint arXiv:2010.11929.

[Durkan et al., 2019] Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. (2019). Neural Spline Flows. Advances in Neural Information Processing Systems, 32:7511–7522.
[Durrett, 2019] Durrett, R. (2019). *Probability: theory and examples*, volume 49. Cambridge university press.

[Germain et al., 2015] Germain, M., Gregor, K., Murray, I., and Larochelle, H. (2015). Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pages 881–889. PMLR.

[Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Nets. *Advances in neural information processing systems*, 27.

[Grathwohl et al., 2018] Grathwohl, W., Chen, R. T., Bettencourt, J., Sutskever, I., and Duvenaud, D. (2018). FFJORD: Free-Form Continuous Dynamics for Scalable Reversible Generative Models. In *International Conference on Learning Representations*.

[Grathwohl et al., 2019] Grathwohl, W., Wang, K.-C., Jacobsen, J.-H., Duvenaud, D., Norouzi, M., and Swersky, K. (2019). Your Classifier is Secretly an Energy Based Model and You Should Treat It Like One. *arXiv preprint arXiv:1912.03263*.

[Ho et al., 2019] Ho, J., Chen, X., Srinivas, A., Duan, Y., and Abbeel, P. (2019). Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pages 2722–2730. PMLR.

[Huang et al., 2018] Huang, C.-W., Krueger, D., Lacoste, A., and Courville, A. (2018). Neural autoregressive flows. In *International Conference on Machine Learning*, pages 2078–2087. PMLR.

[Jaini et al., 2019] Jaini, P., Selby, K. A., and Yu, Y. (2019). Sum-of-squares polynomial flow. In *International Conference on Machine Learning*, pages 3009–3018. PMLR.

[Jang et al., 2016] Jang, E., Gu, S., and Poole, B. (2016). Categorical Reparameterization with Gumbel-Softmax. *arXiv preprint arXiv:1611.01144*.

[Karras et al., 2020] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2020). Analyzing and Improving the Image Quality of Stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *ICLR (Poster)*.

[Kingma and Dhariwal, 2018] Kingma, D. P. and Dhariwal, P. (2018). Glow: Generative Flow with Invertible 1× 1 Convolutions. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10236–10245.

[Kingma et al., 2016] Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. (2016). Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29:4743–4751.
[Kingma and Welling, 2013] Kingma, D. P. and Welling, M. (2013). Auto-encoding Variational Bayes. arXiv preprint arXiv:1312.6114.

[Krizhevsky et al., 2009] Krizhevsky, A., Hinton, G., et al. (2009). Learning Multiple Layers of Features from Tiny Images.

[LeCun, 1998] LeCun, Y. (1998). The MNIST database of handwritten digits. http://yann.lecun.com/exdb/mnist/.

[Liu et al., 2019] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692.

[Liu et al., 2015] Liu, Z., Luo, P., Wang, X., and Tang, X. (2015). Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV).

[Mahajan et al., 2020] Mahajan, S., Bhattacharyya, A., Fritz, M., Schiele, B., and Roth, S. (2020). Normalizing flows with multi-scale autoregressive priors. arXiv preprint arXiv:2004.03891.

[Martin et al., 2001] Martin, D., Fowlkes, C., Tal, D., and Malik, J. (2001). A Database of Human Segmented Natural Images and Its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, volume 2, pages 416–423. IEEE.

[McLachlan and Peel, 2000] McLachlan, G. J. and Peel, D. (2000). Finite Mixture Models. Wiley Series in Probability and Statistics, New York.

[Moon, 1996] Moon, T. K. (1996). The expectation-maximization algorithm. IEEE Signal processing magazine, 13(6):47–60.

[Oliva et al., 2018] Oliva, J., Dubey, A., Zaheer, M., Poczos, B., Salakhutdinov, R., Xing, E., and Schneider, J. (2018). Transformation autoregressive networks. In International Conference on Machine Learning, pages 3898–3907. PMLR.

[Oord et al., 2017] Oord, A. v. d., Vinyals, O., and Kavukcuoglu, K. (2017). Neural Discrete Representation Learning. arXiv preprint arXiv:1711.00937.

[Papamakarios et al., 2017] Papamakarios, G., Pavlakou, T., and Murray, I. (2017). Masked Autoregressive Flow for Density Estimation. In Proceedings of the 31st International Conference on Neural Information Processing Systems, pages 2335–2344.

[Radford et al., 2019] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners.

[Razavi et al., 2019] Razavi, A., van den Oord, A., and Vinyals, O. (2019). Generating Diverse High-Fidelity Images with VQ-VAE-2. In Advances in neural information processing systems, pages 14866–14876.
[Rezende and Mohamed, 2015] Rezende, D. and Mohamed, S. (2015). Variational Inference with Normalizing Flows. In International conference on machine learning, pages 1530–1538. PMLR.

[Rokhlin, 1949] Rokhlin, V. A. (1949). On the Fundamental Ideas of Measure Theory. Matematicheskii Sbornik, 67(1):107–150.

[Salimans et al., 2016] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved Techniques for Training GANs. Advances in neural information processing systems, 29:2234–2242.

[Vogel, 2002] Vogel, C. R. (2002). Computational Methods for Inverse Problems. SIAM.

[Xiao et al., 2017] Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms.
A Proof

A.1 Proof of Example 1 and 2

**Proof.** Assume that the cumulative distribution function (CDF) of Standard Normal distribution is \( \Phi(x) \). Then the CDF for \( X \) is

\[
F(x) = \frac{1}{2} \left[ \Phi \left( \frac{x - \mu}{\sigma} \right) + \Phi \left( \frac{x + \mu}{\sigma} \right) \right]
\]

For case (1), the CDF of \( Z \) is \( \Phi(x) \)

Then, by inverse transform sampling [Vogel, 2002], the optimal transportation function can be written as

\[
G(z) = F^{-1} \circ \Phi(z)
\]

Thus,

\[
\frac{dG(z)}{dz} = \frac{F^{-1}(\Phi(z))}{d\Phi(z)} \cdot \frac{d\Phi(z)}{dz} = \frac{dF^{-1}(\Phi(z))}{d\Phi} \cdot p_Z(z) = \frac{p_Z(z)}{p_X(G(z))}
\]

When \( z = 0 \), \( \Phi(z) = \frac{1}{2} \),

\[
\left. \frac{dG(z)}{dz} \right|_{z=0} = \frac{p_Z(0)}{p_X(0)} = \frac{1}{\sigma} \exp \left( -\frac{\mu^2}{2\sigma^2} \right) = \sigma \exp \left( \frac{\mu^2}{2\sigma^2} \right)
\]

(2) If the CDF of \( Z \) is

\[
\frac{1}{2} \left[ \Phi \left( \frac{x - \mu/2}{\sigma} \right) + \Phi \left( \frac{x + \mu/2}{\sigma} \right) \right]
\]

then similarly,

\[
\left. \frac{dG(z)}{dz} \right|_{z=0} = \frac{p_Z(0)}{p_X(0)} = \exp \left( \frac{\mu^2}{2\sigma^2} - \left( \frac{\mu/2}{2\sigma} \right)^2 \right) = \exp \left( \frac{3\mu^2}{8\sigma^2} \right).
\]

With latent variable, it is quite natural that \( \sigma Z + \mu \sim \mathcal{N}(\mu, \sigma^2) \). \( \square \)

A.2 Proof of Proposition 1 and 2

**Proof.**

\[
\log p_\theta(x) = \log \int p_\theta(x|u) d\pi_\beta(u) = \log \int p_\theta(x|u) \frac{d\pi_\beta(u)}{d\nu_\phi(u)} d\nu_\phi(u) \geq \int \log p_\theta(x|u) d\nu_\phi(u) + \text{KL}(\nu_\phi||\pi_\beta)
\]

(A.1)
By taking $p_\theta(x|u) = p_Z(G_\theta^{-1}(x,u))|\nabla_x G_\theta^{-1}(x,u)|$, the desired loss is obtained.

\[
\text{A.3 Proof of Proposition 3}
\]

Proof. Since $\nu_\phi$ is in the distribution class of $\pi_\beta$, for any $\nu_\phi(u|x)$ and $X$, there exists $\beta_\phi$ such that $\pi_{\beta_\phi} = E_X \nu_\phi(u|x)$.

Thus, by choosing $\beta_\phi$ and taking the expectation over $X$. The loss function becomes,

\[
E_X \ell(\theta, \phi, \beta; x) = -\frac{1}{K} \sum_{k=1}^{K} E_X \log p_Z(G_\theta^{-1}(x,u_{xk})) - \frac{1}{K} \sum_{k=1}^{K} E_X \log |\nabla_x G_\theta^{-1}(x,u)|. \tag{A.2}
\]

Thus, for each $x$, the loss function for $\phi$ and $\theta$ is equivalent to Eq. (3.6).

For optimization of $\beta$, the KL divergence term can also be estimated by Monte Carlo samples. Note that

\[
\text{KL}(\nu_\phi(u|x)\|\pi_\beta(u)) \propto -\int \log \frac{d\pi_\beta(u)}{d\mu_u} d\nu_\phi(u|x) \tag{A.3}
\]

where $\mu_u$ is (1) the Lebesgue measure on $u$ if $u$ is continuous; (2) discrete measure on $u$ if it is discrete or sequential.

Then the Monte Carlo estimation is

\[
-\frac{1}{K} \sum_{i=1}^{K} \log \frac{d\pi_\beta(u_i)}{d\mu_u}, \quad u_i \sim \nu_\phi(u|x). \tag{A.4}
\]
B More discussions

B.1 Latent space $U$

B.1.1 Limited discrete latent space

Discrete latent space is adopted in conventional statistics, especially for clustering analysis. In particular, the idea is to divide the observation data into several groups, each group is associated with one vector: $U = \{v_1, \cdots, v_s\}$. The vector $v_i$ is the representation of each group. This idea is the most basic idea in representation learning that assign a feature vector for each group. The index of each representation in latent space can be recognized as an unsupervised label. However, due to the capacity of discrete latent space, it becomes less popular in current deep learning literature.

For NF framework, the discrete latent space still has significant value. Although the NF framework has explicit likelihood, the introduction of latent variable makes the density computationally available only if $U$ is discrete (Otherwise we need to compute the integral over $u$). Meanwhile, the discrete $u$ is sufficient for constructing a better transportation path in spite of the limited capacity. Thus, for tabular data, the choice of discrete $u$ is ideal to boost conventional NF.

B.1.2 Continuous latent space

The continuous latent space is the most general and widely-used representation recently. Most of feature learning are based on this structure, such as the ResNet pretrained feature, latent space of VAE, etc. For GAN, although we are unable to find $u|x$ explicitly, the design of $u$ is still continuous. In these models, the capacity of continuous $u$ benefits the modeling process. Even if the crucial latent space is discrete one, it is still a subspace of the original continuous latent space. For example, the latent labels can be obtained by continuous $u$ with a linear transformation and a softmax transform.

However, from an NF view, the continuous $u$ loses an important advantage of NF models while brings powerful representation: the painless computation of $p_\theta(x)$. We need to approximate the integral over $\pi_\beta(u)$ to obtain the density form.

B.1.3 Sequential latent space

Another popular latent representation is the sequential ones, i.e., a sequence/tensor of latent representation. $U = \{v_1, \cdots, v_m\}$, where $v_i$ is the representation of each element, $m$ denote the length of sequence. The current space contains $s^m$ states, while plain discrete space only has $s$. Exponentially large state space makes the capacity totally different, so we divide it to a new category. Recently, with the development of transformers, the sequential latent space is particularly becomes a strongly competitor against continuous latent space.

Considering the NF setting, since sequential model is still discrete, the density estimation process can be done with $\nu_\phi(u|x)$ importance sampling. Meanwhile, the space of current $U$ is much larger conventional discrete space, which support more complex data structures such as images, texts, etc. Notably, for this data structure, the $\pi_\beta(u)$ can be highly uneven over states, so we learn the prior from data.
B.2 Experimental Details

All of our experiments are conducted on 11GB NVIDIA 2080Ti GPUs. For toy and tabular data, the encoder architecture is multilayer perceptron (MLP). For visual datasets, the encoder architecture is based on the ResNet and the prior of latent sequence is based on a Transformer. We discuss the experimental details with different datasets respectively.

Toy Data

On toy dataset, we use a sequence of (1) Reverse Permutation; (2) LU Linear Transform; (3) Affine Coupling Transform; for each layer. We have compared the settings of different depth and hidden size of Affine Coupling Transform, such as [4,4], [4,16], [4,32], [10,4], [10,16], [10,32]. The posterior estimator \( \nu \) is approximated by a MLP with a 4 layer 16 hidden nodes.

Tabular Data

On tabular datasets, we follow the architecture and default parameters of neural spline flows with additional context feature encoded by the posterior estimator \( \nu(v|x) \). The variational posterior approximator \( \nu(v|x) \) is a 4-layer MLP encoder with hidden 256 neurons and ReLU activation function. The maximum training epoch is 1000. The model is selected by validation. Adam optimizer is used and the initial learning rate is \( 5 \times 10^4 \).

Visual Data

We first train an encoder-decoder model, where the vocabulary size is 16, embedding dimension is 16, batch size is 32, and learning rate is 1e-5. Then we train a Transformer model, implemented based on a 12-layer, 768-hidden, 12-heads GPT-2 with 512 for maximum input length and 3072 for intermediate size. We initialize it from the pre-trained weights of GPT-2 124M with GPT-2-simple library (https://github.com/minimaxir/gpt-2-simple). The Transformer model is optimized 100,000 steps with AdamW optimizer. The weight decay is set to 0.01 with \( \beta_1 = 0.9, \beta_2 = 0.999 \). The learning rate is 1e-4 and linearly decayed to 0 according to a cosine schedule.

Time Cost Discussion

The time cost vs baseline in (1) Tabular data: is negligible (Sampling: \( 1.001 \times \), Inference: \( 1.007 \times \), 1.0 \times indicates no extra cost). (2) CIFAR-10: is still affordable (Sampling: \( 3.41 \times \), Inference: \( 1.67 \times \)). The computation overhead is similar to larger network. The training of latent prior and NF can be paralleled.