

Variations and Relaxations of Normalizing Flows

Keegan Kelly

NEW YORK UNIVERSITY

KMK9461@NYU.EDU

Lorena Piedras

NEW YORK UNIVERSITY

LP2535@NYU.EDU

Sukrit Rao

NEW YORK UNIVERSITY

STR8775@NYU.EDU

David Roth

NEW YORK UNIVERSITY

DSR331@NYU.EDU

Abstract

Normalizing Flows (NFs) describe a class of models that express a complex target distribution as the composition of a series of bijective transformations over a simpler base distribution. By limiting the space of candidate transformations to diffeomorphisms, NFs enjoy efficient, exact sampling and density evaluation, enabling NFs to flexibly behave as both discriminative and generative models. Their restriction to diffeomorphisms, however, enforces that input, output and all intermediary spaces share the same dimension, limiting their ability to effectively represent target distributions with complex topologies (Zhang and Chen 2021). Additionally, in cases where the prior and target distributions are not homeomorphic, Normalizing Flows can leak mass outside of the support of the target (Cornish et al. 2019; Wu et al. 2020). This survey covers a selection of recent works that combine aspects of other generative model classes, such as VAEs and diffusion, and in doing so loosen the strict bijectivity constraints of NFs to achieve a balance of expressivity, training speed, sample efficiency and likelihood tractability.

Keywords: Generative Modeling, Normalizing Flows, Diffusion

1. Introduction

Research in generative modeling with deep learning has in large part focused on four classes of models: flows, VAEs, diffusion models and GANs. Until recently, GANs had proven the model family capable of producing the highest fidelity generated samples, but a recent string of high-profile successes using diffusion models for natural image (Ho et al. 2020), audio (Kong et al. 2020) and video synthesis (Ho et al. (2022)), trajectory planning (Janner et al. 2022), protein and material design (Luo et al.; Anand and Achim 2022) has called into question their dominance in generative tasks. VAEs on the other hand, are a slightly older class of models that are easier to train but have been less successful at producing realistic data distributions. Some work has gone into improving the expressivity of VAEs (Aneja et al. 2021) but has encountered a tension between VAE expressivity and a tendency towards posterior collapse, where the generated model ignores the latent codes z entirely in favor of learning a capable generator.

This paper presents the fundamentals for each of these basic model classes and a selection of recent works that combine aspects from each to achieve a balance of model expressivity, training speed, sample efficiency and likelihood tractability. In particular, we focus on a selection of papers that loosen the strict bijectivity constraints of Normalizing Flows (NF) and attempt to improve the expressivity and sample efficiency of NFs while retaining as much as possible the likelihood evaluation properties the strict construction affords.

2. Normalizing Flows

Normalizing Flows are notable among the broader family of generative models in that they are not only capable of expressing rich, complex distributions— they are able to do so while also retaining the ability to perform exact density evaluation. They achieve this capacity by expressing a complex target distribution of interest as a bijective, differentiable transformation of a simpler, known base distribution. This formulation provides a learning mechanism using maximum likelihood over i.i.d samples from the target distribution, a sampling mechanism via transformations over points drawn from the base distribution and exact density evaluation using the inverse of the learned transformation and a change of variables with the learned transform’s Jacobian.

Normalizing Flows were popularized in the context of Variational Inference by Rezende and Mohamed (2015) as a choice of tractable posterior for continuous variables that is more capable of representing complex distributions than traditional choices for approximate posteriors, such as Mean Field Approximations. However, the use of flows for density estimation was first formulated by Tabak and Vanden-Eijnden (2010) and was used in subsequent works for clustering and classification tasks in addition to density estimation (Agnelli et al. 2010; Laurence et al. 2014).

The formal structure of a Normalizing Flow is as follows: Let $Z \in \mathbb{R}^D$ be a random variable with known probability density function $p_Z: \mathbb{R}^D \mapsto \mathbb{R}$, referred to as the base distribution and let $X \in \mathbb{R}^D$ be a random variable of interest over which we would like to define a density $p_X: \mathbb{R}^D \mapsto \mathbb{R}$, referred to as the target distribution. We then seek a parameterized transformation $F_\theta: \mathbb{R}^D \mapsto \mathbb{R}^D$ under which $F_\theta(Z) = X$. We restrict our choices for F_θ to bijective, differentiable mappings, known as *diffeomorphisms*. Under these constraints, the density of a point $x \sim X$, can be calculated under a change of variables using the determinant of the transformation’s Jacobian, J_F , as follows:

$$p_X(x) = p_Z(z)|\det J_F(z)|^{-1}$$

or, framed in terms of the reverse direction,

$$p_X(x) = p_Z(F_\theta^{-1}(x))|\det J_F^{-1}(x)|.$$

This product represents the probability density of the inverse-transformed point in the base distribution multiplied by the change in volume incurred by the transformation in an infinitesimal neighborhood around z . In practice, F_θ is often constructed as the composition of a sequence of N diffeomorphisms $f_{1,\theta_1}, \dots, f_{M,\theta_M}$ such that

$$F_\theta = f_{1,\theta_1} \circ \dots \circ f_{M,\theta_M}$$

Since each of these sub-transformations is itself invertible, their composition is also invertible and bijective. The determinant of J_F can be computed exactly as:

$$\det J_F(z) = \prod_{i=1}^N \det J_{f_i, \theta_i}$$

and the function's inverse as

$$F_\theta^{-1} = f_{M, \theta_M}^{-1} \circ \cdots \circ f_{1, \theta_1}^{-1}.$$

2.1 Training of Normalizing Flows

Normalizing Flows can be trained in one of two ways, depending on the nature of access to the target distribution during training. In the setting where samples from p_x are available, but not their densities, the model parameters θ can be estimated using the forward KL-Divergence:

$$\begin{aligned} \mathcal{L}_\theta &= D_{KL} [p_x^*(x) \parallel p_X(x; \theta)] \\ &= -\mathbb{E}_{p_x^*(x)} [\log p_x(x; \theta)] + \text{const.} \\ &= -\mathbb{E}_{p_x^*(x)} [\log p_Z(F_\theta^{-1}(x)) + \log |\det J_F^{-1}(x)|] + \text{const.} \end{aligned}$$

With a set of N samples $\{x_i\}_{i=1}^N$, we can estimate the above loss as

$$\mathcal{L}_\theta \approx -\frac{1}{N} \sum_{i=1}^N \log p_Z(F_\theta^{-1}(x_i)) + \log |\det J_F^{-1}(x_i)| + \text{const.}$$

In the setting where it is possible to evaluate the target density p_x^* cheaply, but it is not straightforward to draw samples from said distribution, model parameters θ can be estimated using the reverse KL-Divergence:

$$\begin{aligned} \mathcal{L}_\theta &= D_{KL} [p_X(x; \theta) \parallel p_x^*(x)] \\ &= \mathbb{E}_{p_X(x; \theta)} [\log p_X(x; \theta) - \log p_x^*(x)] \end{aligned}$$

2.2 Limitations of Normalizing Flows

Though Normalizing Flows are in principle capable of representing arbitrarily complex target distributions (Papamakarios et al. 2021), for choices of simple base distributions and reasonably smooth transformations they suffer from topological limitations (Stimper et al. 2021). Strict bijectivity enforces that the input, output and all intermediary spaces share identical dimensionality and topology. Cornish et al. (2019) demonstrate that for base and target distributions with distinct support topologies (e.g differing in number of connected components, number of holes), and choice of candidate transformation where F_θ and F_θ^{-1} are continuous, it is impossible to represent the target distribution as a transformation of the base distribution and an arbitrarily accurate approximation requires the bi-Lipschitz

constant of F_θ , a measure of a function’s ”invertibility” (Behrmann et al. 2020) to approach ∞ .

Evidence of this limitation can be seen in a ”smearing” effect when attempting to represent a bi-modal or multi-modal target distribution using a standard unimodal Gaussian as a base distribution, where sharp boundaries cannot be expressed and density is leaked outside the support of the true target distribution. (Figure 1) Further, under the *manifold hypothesis* (Bengio et al. 2012), if real-world distributions reside in a low-dimensional ($r \ll d$) manifold of the spaces they inhabit, it is a relative certainty that the base and target distributions will have mismatched support topologies and that probability density will leak outside of the target support.

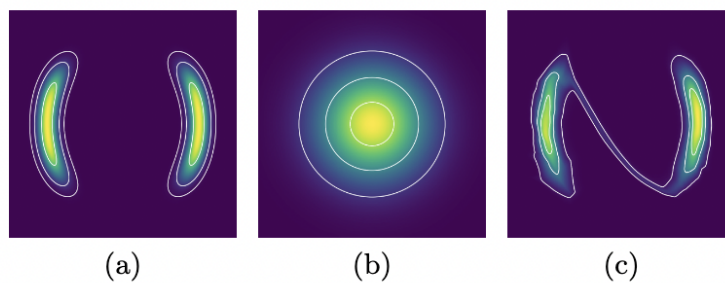


Figure 1: An example of ”smearing” in (c), where the target distribution (a) and the base distribution (b) differ in their number of connected components.

Figure taken from Stimper et al. (2021)

3. Variational Autoencoders

Variational Autoencoders (VAEs) are a likelihood-based class of models that provide a principled framework for optimizing latent variable models (Kingma and Welling 2013). It consists of two models- a recognition model or encoder and a generative model or decoder that are coupled together. The recognition model approximates the posterior over the latent random variables which is passed as input to the generative model to generate samples. The generative model, on the other hand, provides a scaffolding or structure for the recognition model to learn meaningful representations of the data. The recognition model is the approximate inverse of the generative model according to Bayes’ rule. (Kingma and Welling 2019)

In the typical setting for a latent variable model, we have some observed variables and some unobserved variables. To estimate the unconditional density of the observed variables, also called the model evidence, we marginalize over the joint distribution of the observed and unobserved variables, parameterized by θ . This is given by

$$p_\theta(x) = \int_Z p_\theta(x, z) dz$$

Framing the problem through an implicit distribution over x provides a great deal of flexibility. When we marginalize over the latents we end up with a compound probability distribution or mixture model. For example, if z is discrete and $p_\theta(x|z)$ is a Gaussian distribution, then $p_\theta(x)$ will be a mixture of Gaussians. For continuous z , $p_\theta(x)$, can be seen as an infinite mixture. Thus, depending on the choice of the latent distribution, we can control the expressivity of the unconditional density $p_\theta(x)$, as desired.

This compound distribution, however, is obtained by integrating over the support of the latent distribution. Most of the time, this integral is intractable and thus, we cannot differentiate with respect to its parameters and optimize it using gradient descent. While the joint density $p_\theta(x, z)$ is efficient to compute, the intractability of $p_\theta(x)$, is related to the intractability of the posterior over the latent variable, $p_\theta(z|x)$ (Kingma and Welling 2019). From the chain rule, we have the following relationship between the densities

$$p_\theta(z|x) = \frac{p_\theta(x, z)}{p_\theta(x)}$$

The intractability of $p_\theta(z|x)$ leads to the intractability of $p_\theta(x)$. To overcome this hurdle, we employ approximate inference techniques. The framework of VAEs provides a computationally efficient way for optimizing latent variable models jointly with a corresponding inference model using gradient descent (Kingma and Welling 2019). This is achieved by introducing the encoder or recognition model- a parametric inference model $q_\phi(z|x)$, where ϕ is the set of variational parameters.

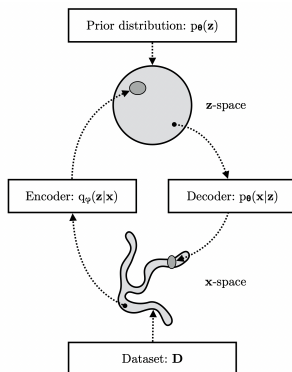


Figure 2: Computational flow in a VAE
Figure taken from Kingma and Welling (2019)

Consequently, the optimization objective of VAEs is the variational lower bound or evidence lower bound (ELBO), where we optimize the variational parameters ϕ such that

$$q_\phi(z|x) \approx p_\theta(z|x)$$

It follows from the derivation shown below

$$\begin{aligned}
\log p_\theta(x) &= \mathbb{E}_{q_\phi(z|x)} \log p_\theta(x) \\
&= \mathbb{E}_{q_\phi(z|x)} \log \left[\frac{p_\theta(x, z)}{p_\theta(z|x)} \right] && \text{(chain rule)} \\
&= \mathbb{E}_{q_\phi(z|x)} \log \left[\frac{p_\theta(x, z) q_\phi(z|x)}{q_\phi(z|x) p_\theta(z|x)} \right] \\
&= \underbrace{\mathbb{E}_{q_\phi(z|x)} \log \left[\frac{p_\theta(x, z)}{q_\phi(z|x)} \right]}_{=\mathcal{L}_{\phi, \theta}(x) \text{ (ELBO)}} + \underbrace{\mathbb{E}_{q_\phi(z|x)} \log \left[\frac{q_\phi(z|x)}{p_\theta(z|x)} \right]}_{=D_{KL}(q_\phi(z|x)||p_\theta(z|x))}
\end{aligned}$$

The second term is the Kullback-Leibler (KL) divergence between $q_\phi(z|x)$ and $p_\theta(z|x)$, while the first term is the variational lower bound or evidence lower bound (ELBO).

Since the KL divergence is non-negative, the ELBO is the lower bound on the log-likelihood of the data

$$\begin{aligned}
\mathcal{L}_{\phi, \theta}(x) &= \log p_\theta(x) - D_{KL}(q_\phi(z|x)||p_\theta(z|x)) \\
\mathcal{L}_{\phi, \theta}(x) &\leq \log p_\theta(x)
\end{aligned}$$

Thus, we can observe that maximizing the ELBO $\mathcal{L}_{\phi, \theta}(x)$ with respect to θ and ϕ , will have the following consequences

- It will approximately maximize the marginal likelihood $p_\theta(x)$, implying that our generative model will get better
- It will minimize the KL divergence between $q_\phi(z|x)$ and $p_\theta(z|x)$, implying our approximation of the posterior, $q_\phi(z|x)$, will get better

4. Denoising Diffusion

Diffusion-based generative models are parameterized Markov chains mainly used to create high quality images and videos, and also utilized in data compression and representation learning on unlabeled data. Diffusion models are both tractable and flexible, making them easier to train, evaluate and sample from. The transitions of the Markov chain gradually add noise to the data and then learn to reverse the diffusion process, producing desired samples after a finite time. Unlike VAEs, diffusion models are latent variable models where the dimensionality of latent space is the same as the original data. The idea of using diffusion for a generative process was initially introduced by Sohl-Dickstein et al. (2015) in 2015. Song and Ermon (2019) and Ho et al. (2020) improved the initial approach a couple of years after. The latter showed that diffusion models were capable of generating high quality images and unveiled an equivalence with denoising score matching in training and Langevin dynamics at the sampling stage.

The forward diffusion process gradually adds a small amount of noise in T steps to the original data until it is indistinguishable. A variance schedule β_1, \dots, β_T , where $w_i \in (0, 1)$, is used to regulate the size of each step. If the noise is small enough, the transitions of the

reverse diffusion process will be conditional Gaussians as well. Given a point sampled from the original data distribution $x_0 \sim q(x)$ we have the following transition probabilities Weng (2021):

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t I)$$

The forward process in a diffusion probabilistic model is fixed, other diffusion models such as diffusion normalizing flows have a trainable forward process Zhang and Chen (2021). A desirable property of the forward process shown by Sohl-Dickstein et al. (2015) is that we can sample x_t given x_0 at any time step without having to apply q repeatedly

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t)I)$$

with $\alpha_t := 1 - \beta_t$ and $\bar{\alpha}_t := \prod_{s=1}^t \alpha_s$.

We could use $q(x_{t-1}|x_t)$ to revert the forward diffusion process and generate a sample from the real distribution using random noise as input. Unfortunately $q(x_{t-1}|x_t)$ is intractable, therefore we will learn a model $p_\theta(x_{t-1}|x_t)$ to approximate it. Notably, the reverse conditional probability is tractable if we condition on x_0 Weng (2021). Similar to VAE, we can use a variational lower bound to optimize $-\log p_\theta(x_0)$. After rewriting the lower bound into several KL-divergence terms and one entropy term and ignoring all terms that don't have any learnable parameters, we get two components $L_0 = -\log p_\theta(x_0|x_1)$ and $L_t = D_{KL}(q(x_t|x_{t+1}, x_0)||p_\theta(x_t|x_{t+1}))$. L_t is the KL divergence of two Gaussian distributions, where $q(x_t|x_{t+1}, x_0)$ is the tractable reverse conditional distribution mentioned earlier. In Ho et al. (2020), L_0 is modeled using a separate discrete decoder and a fixed variance term.

5. Score-Based Methods

Transport models employ maximum likelihood estimation to learn probability distributions. This reliance can pose a major challenge given complex partition functions, which may prove intractable. Some models add constraints to ensure MLE is feasible; the bijectivity of Normalizing Flows and the approximation of the partition function in VAEs are two such methods to overcome intractability. Another framework to address this scenario is known as the score-based method. For this setup, we model the score function rather than the density function directly:

$$s_\theta(x) = \nabla_x \log p_\theta(x)$$

Partition function Z_θ is reduced to zero in the context of $\nabla_x \log Z_\theta$ and thus is independent of the score-based model. We are therefore able to sidestep any challenging computation posed by the partition function while training. This setup introduces flexibility, where we can now work with many families of models that may have been otherwise intractable.

Score-based diffusion is an extension upon this method. As in the previous section on diffusion, this model class involves both a forward and backward stochastic differential equation. Again, the forward pass returns a noisy distribution:

$$x_t = e^{-t}x + \sqrt{1 - e^{-t}}z$$

Where $x \sim \pi_d$ and $z \sim N(0, I)$.

In score-based diffusion, the reversed pass can now be written as a flow composed of diffusion drift plus an exact score:

$$dY_t = [Y_t - \nabla \log \pi_t(Y_t)]dt + \sqrt{2}dB_t$$

Where $Y_t = X_{T-t}$ (Bruna (2022)).

The challenge now falls on estimating the scores from the data. This is particularly impactful in low density regions, where there is less data available to compute scores. In such cases, the model may produce poor quality sampling. To work around this obstacle, noise can be added to the data in increasingly larger magnitudes so that the model can at first train on less corrupted data, then learn low data density regions as the magnitude of noise grows. In this way, adding noise adds stability to score-based methods and aids in producing higher quality samples (Song and Ermon 2019).

Score-based models can also be used to compute exact likelihoods. This requires converting the reverse stochastic differential equation into an ordinary differential equation, as seen below:

$$dx = [f(x, t) - \frac{1}{2}g^2(t)\nabla_x \log p_t(x)]dt$$

The above equation, known as the probability flow ODE, becomes a representation of a neural ODE when the score function $\nabla_x \log p_t(x)$ is approximated by the score-based model $s_\theta(x, t)$. Because of this relationship, the probability flow ODE takes on characteristics of a neural ODE, such as invertibility, and can compute exact likelihoods using the instantaneous change-of-variables formula (Song et al. 2021).

6. Relaxing Constraints

In this section, we explore several works that formulate new model classes by relaxing the strict bijectivity constraints of Normalizing Flows. These works expand the family of admissible functions to include surjective/stochastic transformations and take inspiration from score-based models and diffusion by introducing noise into the training process.

6.1 SurVAE Flows

In an attempt to place VAEs and Normalizing Flows in a common context, Nielsen et al. (2020) introduce SurVAE Flows— a class of models composed of *surjective transformations*, allowing for models that mix bijective, surjective and stochastic components in a single end-to-end trainable framework. They identify three mechanisms needed for probabilistic generative models in this family:

1. A forward transformation: $p(x|z)$
2. An inverse transformation: $p(z|x)$
3. A likelihood contribution: $p(x)$

In a normalizing flow, the forward transformation and reverse transformations are deterministic and can be represented as $p(x|z) = \delta(x - F(z))$ and $p(z|x) = \delta(z - F^{-1}(x))$. In a VAE, both directions are stochastic and a variational approximation $q(z|x)$ is used in place of the intractable posterior.

They use this decomposition to draw formal connections between stochastic transformations (VAEs) and bijections (normalizing flows) using Dirac delta functions. In particular, they show that the marginal density $p(x)$ can be expressed under both paradigms as:

$$\log p(x) \simeq \log p(z) + \mathcal{V}(x, z) + \mathcal{E}(x, z)$$

where $\mathcal{V}(x, z)$ represents the likelihood contribution and $\mathcal{E}(x, z)$ represents the 'looseness' of the provided bound. Under VAEs and other stochastic transformations, the likelihood contribution term is calculated as $\log \frac{p(x|z)}{q(z|x)}$ and the 'bound looseness' term is calculated as $\log \frac{q(z|x)}{p(z|x)}$, while under normalizing flows and other bijections, the likelihood contribution term is $\log |\det J|$ and the 'bound looseness' term is 0.

Through the use of surjective, non-injective layers, the authors present constructions that allow for *inference surjections*— models with exact inverses and stochastic forward transformations— and *generative surjections*— models with exact forward transformations and stochastic right inverses. In doing so, they formulate models that bypass the dimensionality constraints enforced by bijectivity without sacrificing the ability to perform exact likelihood evaluation.

The surjective layers they introduce include absolute value, max-value and stochastic permutation, which they use to demonstrate strong experimental results on synthetic modeling tasks. They demonstrate the effectiveness of surjective layers on a handful of synthetic modeling tasks, particularly those with inherent symmetries. Importantly for this survey, these experiments also demonstrate an ability to model sharper boundaries than a fully bijective flow is capable of producing.

The authors argue that a number of recently proposed generative model types can be understood as SurVAE flows, including diffusion models (Ho et al. (2020)), continuously indexed normalizing flows (Cornish et al. (2019)), stochastic normalizing flows (Wu et al. (2020)) and normalizing flows acting on augmented data spaces (Huang et al. (2020)).

6.2 Stochastic Normalizing Flows

Stochastic Normalizing Flows (SNF) are a generalization of the Normalizing Flow framework introduced by Wu et al. (2020). They offer certain benefits over classical stochastic sampling methods and Normalizing Flows for sampling from a known energy model specified up to a normalization constant. Sampling methods such as Markov Chain Monte Carlo (MCMC) or Langevin Dynamics (LD) may have trouble converging because of slow mixing times and local energy minima— adding a deterministic transformation can help alleviate this problem. On the other hand, introducing noise to relax Normalizing Flow's bijectivity constraints can help solve the topological constraints mentioned in 2.2. In Figure 3 we show the double-well example, by adding stochasticity we're able to successfully separate the modes of the distributions avoiding the "smearing" effect.

Similar to NFs, SNFs are a sequence of deterministic transformations. Their contribution comes from adding stochastic blocks, such as Langevin, Metropolis-Hastings, VAE,

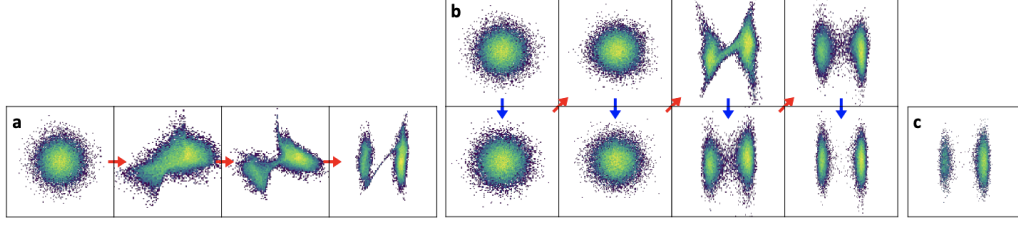


Figure 3: Double well problem: a) Normalizing flows, b) NF with stochasticity, c) Sample from true distribution

Figure taken from Wu et al. (2020)

and diffusion normalizing flow layers. Both the deterministic and stochastic transformations help modify a prior into a complicated target distribution. We can use KL divergence to train NFs and SNFs. In the former we can calculate the probability density to generate a sample $p_x(x)$ using change of variables, however, we can no longer do so— with the introduction of stochasticity, SNFs are no longer invertible. As described in 2, we can train a Normalizing Flow by energy-based training, used when we have a model for the target energy, or maximum likelihood training, when we have samples. We need to generalize the notion of energy and maximum likelihood training in order to train an SNF. We start by defining $\mu_z(x) \propto e^{-u_z(z)}$ as our latent space distribution, $p_x(x) \propto e^{-u_x(x)}$ as our output distribution, and maximizing the importance weights

$$\log w(z \rightarrow x) = \log \frac{\mu_x(x)}{p_x(x)} = -u_x(x) + u_z(z) + \sum_t \nabla S_t(y_t)$$

where $y_{t+1}|y_t \sim q_t(y_t \rightarrow y_{t+1})$, $y_t|y_{t+1} \sim \tilde{q}_t(y_{t+1} \rightarrow y_t)$ are the forward/backward probability distributions at t , we no longer have deterministic flow layers, and $\nabla S_t = \log \frac{\tilde{q}_t(y_{t+1} \rightarrow y_t)}{q_t(y_t \rightarrow y_{t+1})}$ represents the forward-backward probability ratio of step t . By maximizing the importance weights we get the following expressions for energy base training

$$\min \mathbb{E}_{\mu_z(x) \mathbb{P}_f(z \rightarrow x)} [-\log(w(z \rightarrow x))] = \text{KL}(\mu_z(x) \mathbb{P}_f(z \rightarrow x) || \mu_x(x) \mathbb{P}_b(x \rightarrow z))$$

and for maximum likelihood training

$$\min \mathbb{E}_{\mu_x(x) \mathbb{P}_b(x \rightarrow z)} [-\log(w(z \rightarrow x))] = \text{KL}(\mu_x(x) \mathbb{P}_b(x \rightarrow z) || \mu_z(x) \mathbb{P}_f(z \rightarrow x)).$$

where $\mu_z(x) \mathbb{P}_f(z \rightarrow x)$, $\mu_x(x) \mathbb{P}_b(x \rightarrow z)$ are our forward and backward pass probabilities. Notably, the KL divergence of the paths is an upper bound to the KL divergence of the marginal distributions.

$$\text{KL}(p_x || \mu_x) \leq \text{KL}(\mu_z(x) \mathbb{P}_f(z \rightarrow x) || \mu_x(x) \mathbb{P}_b(x \rightarrow z))$$

Finally, we can draw asymptotically unbiased samples from our target distribution $x \sim \mu_x(x)$ by employing the Metropolis-Hastings algorithm and using the importance weights shown above.

6.3 Diffusion Normalizing Flow

Diffusion Normalizing Flow (Zhang and Chen (2021)), or DiffFlow, was introduced as a cross between Normalizing Flows and Diffusion models. DiffFlow is composed of two neural stochastic differential equations (SDEs): a forward pass F that transforms the data X into a simple distribution like Gaussian and a backward pass B that removes noise from the simple distribution to generate samples that match the target data distribution. Like diffusion models, the SDEs are jointly trained to minimize the KL divergence between the forward pass distribution and the backward pass distribution at final time τ . The objective equation is as follows:

$$KL(p_{F(\tau)}|p_{B(\tau)}) = \mathbb{E}_{\tau \sim p_F}[\log p_F(x_0)] + \mathbb{E}_{\tau \sim p_F}[-\log p_B(x_N)] + \sum_{i=1}^{N-1} \mathbb{E}_{\tau \sim p_F}[-\log \frac{p_F(x_i|x_{i-1})}{p_B(x_{i-1}|x_i)}]$$

Similar to Normalizing Flows, DiffFlow is able to learn while mapping to the latent space. However, DiffFlow relaxes the bijectivity constraint of NFs on this mapping. In doing so, DiffFlow has more expressivity and can learn distributions with sharper boundaries. Further, bijectivity may prevent models from having density support over the whole space. Thus in lifting the constraint, DiffFlow has been found to perform better on tasks like image generation of complex patterns. The authors also claim that the boosted expressivity of DiffFlow results in better performance in likelihood over other NF implementations(Zhang and Chen (2021)).

Diffusion Normalizing Flow bypasses the bijectivity constraint by adding noise to the forward stochastic differential equation. Most diffusion models add noise indiscriminately, which can require many iterations to reach Gaussian noise and can lead to generated distributions with corrupted or missing details. On the other hand, due to the trainability of the forward SDE, DiffFlow adds noise only to targeted areas. Thus, DiffFlow can diffuse noise more efficiently and retain topological details that might have been blurred out in other diffusion processes.

Similar to diffusion, DiffFlow SDEs are composed of a drift term f , a vector valued function, and a diffusion term g , a scalar valued function. The equations are as follows:

$$\text{Forward SDE: } dx = f(x, t, \theta)dt + g(t)dw$$

$$\text{Backward SDE: } dx = [f(x, t, \theta) - g^2(t)s(x, t, \theta)]dt + g(t)dw,$$

where x is data at time t and w represents the standard Brownian motion. The main distinguishing factor from Diffusion models is that DiffFlow includes the θ parameter in the drift term which makes the SDEs learnable. From these equations, it is clear that when the diffusion term g tends to 0, DiffFlow reduces to Normalizing Flows.

Given the SDEs above, the discretized equations can be written as:

$$\begin{aligned} x_{i+1} &= x_i + f_i(x_i)\Delta t_i + g_i\delta_i^F\sqrt{\Delta t_i} \\ x_i &= x_{i+1} - [f_{i+1}(x_{i+1}) - g_{i+1}^2(x_{i+1})]\Delta t_i + g_{i+1}\delta_i^B\sqrt{\Delta t_i} \end{aligned}$$

Returning to KL divergence, given that the first term is a constant and utilizing the discretized SDEs, the objective can be reduced to the form:

$$L = \mathbb{E}_{\delta^F; x_0 \sim p_0} [-\log p_B(x_N) + \sum_{i=1} \frac{1}{2} (\delta_i^B(\tau))^2]$$

where noise is represented as:

$$\delta_i^B(\tau) = \frac{1}{g_{i+1}\sqrt{\Delta t}} [x_i - x_{i+1} + [f_{i+1}(x_{i+1}) = g_{i+1}^2 s_{i+1}(x_{i+1})] \Delta t]$$

Loss can now be minimized with Monte Carlo gradient descent (Zhang and Chen (2021)).

6.4 Stochastic Normalizing Flows and Diffusion Normalizing Flows

Zhang and Chen (2021) introduced Diffusion Normalizing Flows (DNF) as a new type of model. Nevertheless, per Hagemann et al. (2021) if we view SNMs as a pair of Markov chains $((X_0, \dots, X_t), (Y_t, \dots, Y_0))$ where (Y_t, \dots, Y_0) is the reverse Markov chain of (X_0, \dots, X_t) , we can view DNFs as a type of SDFs with specific forward and backward layers

$$\begin{aligned} \mathcal{K}_t(x, \cdot) &= P_{X_t|X_{t-1}=x} = \mathcal{N}(x + \epsilon g_{t-1}(x), \epsilon h_{t-1}^2) \\ \mathcal{R}_t(x, \cdot) &= P_{Y_{t-1}|Y_t=x} = \mathcal{N}(x + \epsilon(g_t(x) - h_t^2 s_t(x)), \epsilon h_t^2) \end{aligned}$$

The equations above come from the Euler discretization with step size ϵ of the stochastic differential equation with drift g_t , diffusion coefficient h_t and Brownian motion B_t

$$dX_t = g_t(X_t)dt + h_t dB_t$$

7. Discussion

In this section, we talk about the role of stochasticity in normalizing flows and compare the various techniques introduced above on the basis of the following criteria:

- *Expressivity*- while expressivity is usually used in a broad sense in the literature, we focus on each technique's ability to capture the various modes of the distribution they are trying to model as well as regions with relatively low density.
- *Training speed*- we characterize training speed as the time taken by each technique to reach convergence.
- *Ease of likelihood computation*- for this criterion, we look at the tractability of the likelihood computation for density estimation.
- *Sampling efficiency*- we differentiate sampling efficiency from data efficiency, with the former referring to the computational cost required to generate samples with the latter referring to the number of samples required for optimization.

We also direct the reader to the comprehensive comparison of the bulk of the techniques covered in this paper performed by Bond-Taylor et al. (2022)

7.1 Expressivity

As described in section 3, VAEs employ the use of latent variables. The choice of these provides them with a great deal of flexibility, resulting in highly expressive models. On the other hand, bijectivity constraints imposed by the normalizing flows framework result in representational insufficiency. Their representational capacity depends on the type of flow used in the model. For example, linear flows are limited in their expressivity. On the other hand, coupling and autoregressive flows, two of the most widely used flow architectures, enable normalizing flows to represent very complex distributions. However, they are still limited in their expressivity due to the invertibility constraint imposed by the framework (Kobyzev et al. 2021).

Stochastic normalizing flows overcome some of these limitations by incorporating stochastic sampling blocks into the normalizing flow framework, thus improving representational capacity over deterministic flow architectures by overcoming topological constraints (Wu et al. 2020). DiffFlow enjoys better expressivity than vanilla normalizing flows by overcoming their bijectivity constraints by adding noise. The aforementioned constraints prevent normalizing flows from expanding density support to the whole space when transforming complex distributions to a base distribution. As a result, DiffFlow can learn distributions with sharper boundaries (Zhang and Chen 2021).

Score-based methods are notably flexible due to the fact that they are independent of the normalizing constant Z_θ . This allows score-based methods to represent a more diverse set of models. Similar to other types of models, score-based methods are limited by the constraint that the dimension between their input and output must match. Otherwise, score-based models may take the form of any vector-valued function and thus are quite expressive (Song and Ermon 2019).

Surjective flows empirically demonstrate an ability to represent sharper boundaries than vanilla NFs, however, their methods are non-general and require prior knowledge of relevant symmetries in the target distribution (Nielsen et al. 2020).

7.2 Training speed

Normalizing Flows are known to be inefficient and difficult to train due to invertibility constraints on transformations and as a consequence input and output spaces with the same dimension (Bond-Taylor et al. 2022). By adding noise and bypassing strong bi-Lipschitz limitations, stochastic normalizing flows are easier to optimize. Moreover, adding stochastic layers is not computationally costly since they have linear computational complexity Wu et al. (2020).

DiffFlow tends to train slower in comparison to other models. While certain aspects, such as a trainable forward function, help improve efficiency, DiffFlow ultimately relies on backpropagation, making it slow to train. On the other hand, VAEs reach convergence quite quickly. Due to the reparameterization trick proposed by Kingma and Welling (2013), VAEs can use SGD during optimization.

Score-based models may struggle with training for low density regions, especially if the target distribution has multiple modes with a degree of separation. The model may then fail to converge in a reasonable time. As mentioned in section 5, adding progressively more noise to the data in training can improve model convergence in such cases.

7.3 Ease of Likelihood Computation

Normalizing flows benefit from having bijective and invertible (diffeomorphisms) transformations applied to base distributions, resulting in the ability to compute exact likelihoods Kobyzev et al. (2021). Adding noise to stochastic and diffusion normalizing flows increases expressivity over normalizing flows but at the cost of not being able to compute exact likelihoods. The parameters of a stochastic normalizing flow can be optimized by minimizing the KL divergence between the forward and backward path probabilities. This minimization makes use of the variational approximation, which precludes them from computing exact likelihoods Wu et al. (2020). Diffusion normalizing flows add noise in the forward stochastic differential equation. Consequently, they use the reparameterization trick proposed by Kingma and Welling (2013) and thus we cannot compute exact likelihoods. To estimate likelihoods they use the marginals distribution equivalent SDE Zhang and Chen (2021).

VAEs optimize the variational lower bound, an approximation of the log-likelihood we are trying to optimize and as a result, we cannot compute exact likelihoods. Importance sampling or Monte Carlo sampling techniques are used to compute the likelihood of data after training is completed Kingma and Welling (2019). Finally, score-based methods provide an avenue to compute exact likelihoods. This requires some manipulation of the equations and the introduction of invertibility into the model. According to Song et al. (2021), score-based methods are then able to achieve ‘state-of-the-art likelihoods’ on some image generation tasks.

Among the transformations proposed in Nielsen et al. (2020), only inference surjections, i.e. surjective layers that have full support in the base distribution and partial support in the target distribution, are able to produce exact likelihoods. Generative surjections, on the other hand, can only provide stochastic lower bound estimates.

7.4 Sampling Efficiency

Sampling efficiency is mainly affected by the complexity of the model and number of iterations to generate a sample. For example, VAEs consist of an encoder and decoder that are typically complex neural networks. On the other hand, VAEs can generate samples from a single network pass and are thus more efficient than other energy-based models such as stochastic normalizing flows Bond-Taylor et al. (2022).

The sampling efficiency of normalizing flows is related to the cost of the generative direction. However, since the transformations applied to the base distribution are deterministic, samples can be generated in a single network pass and thus, normalizing flows enjoy high sampling efficiency. In comparison, diffusion normalizing flows have poor sampling efficiency, since they require MCMC during sampling. Nevertheless, they have better sampling efficiency than diffusion probabilistic models since they require fewer discretization steps Zhang and Chen (2021). Similar to diffusion normalizing flows, stochastic normalizing flows have lower sampling efficiency than vanilla normalizing flows because they use an MCMC method, Metropolis-Hastings algorithm, to generate samples.

Score-based methods tend to be slow in generating samples, due to the iterative nature of their sampling process. However, score-based methods are often able to produce high quality samples, comparable to GANs in image generation (Song and Ermon (2019)).

7.5 On the Role of Stochasticity

Both Stochastic Normalizing Flows and Diffusion Normalizing Flows introduce stochasticity into their model formulations, though they provide different explanations for the role that stochasticity plays in improving expressivity. Wu et al. (2020) frame the addition of stochastic layers in SNFs as incorporating the strengths of deterministic methods at performing large-scale probability transport with the fine-grained expressivity of Metropolis-Hasting MC sampling—effectively removing samples in areas of lower probability density without incurring the sampling time costs of running a fully MC-reliant model (Wu et al. 2020). Zhang and Chen (2021), on their other hand, attribute the expressivity improvements by DNFs to an expansion of the training support to larger areas of the ambient space, improving gradient coverage for training (Zhang and Chen 2021).

Both agree that adding stochasticity is central to bypassing topological constraints and representing sharp density boundaries in the target space, but the exact mechanism by which it improves expressivity is not fully elucidated by either work. Though beyond the scope of this paper, Bansal et al. (2022) demonstrates experimental evidence of successful diffusion-like models trained using deterministic, non-Gaussian forward processes, such as blurring and masking, calling into question the need for stochastic noise at all. None of the surjective layers proposed Nielsen et al. (2020) utilize added noise, yet they are nonetheless able to represent sharp boundaries in the target distribution. The role and necessity of added noise in improving model expressivity are not clear from these works and require further investigation.

8. Conclusion

This paper delved into a variety of generative models and compared the relative performance of each on expressivity, training speed, sample efficiency and likelihood tractability. Starting from a basis of likelihood-based models, we explored the ability of Normalizing Flows and Variational Autoencoders to directly learn a distribution’s probability density in addition to their capacity to generate samples. VAEs have an encoder-decoder framework trained by optimizing the evidence lower bound, while NFs are structured as a series of bijective differentiable transformations on a data distribution to a simple base distribution.

The strict constraints of the NF architecture narrow the types of distributions that the model can represent. Thus, we explored several models that relaxed the strict bijectivity constraint of NFs. The variations we studied borrowed aspects from different frameworks, including diffusion and score-based models, and introduced stochasticity into the training process. The introduction of noise adds both flexibility and stability to these models. The variations of NFs have performed well in practice, particularly on sampling tasks like image generation. While they cannot be used to compute exact likelihoods, they add much to the field in terms of expressivity and sampling efficiency.

References

- J. Agnelli, Martin Cadeiras, E. Tabak, Turner Cristina, and Eric Vanden-Eijnden. Clustering and classification through normalizing flows in feature space. *Multiscale Modeling Simulation*, 8:1784–1802, 01 2010. doi: 10.1137/100783522.
- Namrata Anand and Tudor Achim. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models, 2022. URL <https://arxiv.org/abs/2205.15019>.
- Jyoti Aneja, Alex Schwing, Jan Kautz, and Arash Vahdat. A contrastive learning approach for training variational autoencoder priors. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 480–493. Curran Associates, Inc., 2021. URL <https://proceedings.neurips.cc/paper/2021/file/0496604c1d80f66fbeb963c12e570a26-Paper.pdf>.
- Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie S. Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise, 2022. URL <https://arxiv.org/abs/2208.09392>.
- Jens Behrmann, Paul Vicol, Kuan-Chieh Wang, Roger B. Grosse, and Jörn-Henrik Jacobsen. On the invertibility of invertible neural networks, 2020. URL <https://openreview.net/forum?id=BJlVeyHFwH>.
- Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives, 2012. URL <https://arxiv.org/abs/1206.5538>.
- Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G. Willcocks. Deep generative modelling: A comparative review of VAEs, GANs, normalizing flows, energy-based and autoregressive models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7327–7347, nov 2022. doi: 10.1109/tpami.2021.3116668. URL <https://doi.org/10.1109%2Ftpami.2021.3116668>.
- Joan Bruna. Lecture notes from inference and representation - ds-ga 1005, December 2022.
- Rob Cornish, Anthony L. Caterini, George Deligiannidis, and Arnaud Doucet. Relaxing bijectivity constraints with continuously indexed normalising flows, 2019. URL <https://arxiv.org/abs/1909.13833>.
- Paul Hagemann, Johannes Hertrich, and Gabriele Steidl. Generalized normalizing flows via markov chains, 2021. URL <https://arxiv.org/abs/2111.12506>.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL <https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf>.

- Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models, 2022. URL <https://arxiv.org/abs/2204.03458>.
- Chin-Wei Huang, Laurent Dinh, and Aaron Courville. Augmented normalizing flows: Bridging the gap between generative flows and latent variable models, 2020. URL <https://arxiv.org/abs/2002.07101>.
- Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis, 2022. URL <https://arxiv.org/abs/2205.09991>.
- Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013. URL <https://arxiv.org/abs/1312.6114>.
- Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *CoRR*, abs/1906.02691, 2019. URL <http://arxiv.org/abs/1906.02691>.
- Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, nov 2021. doi: 10.1109/tpami.2020.2992934. URL <https://doi.org/10.1109/tpami.2020.2992934>.
- Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.
- Peter Laurence, Ricardo J Pignol, and Esteban G Tabak. Constrained density estimation. In *Quantitative Energy Finance*, pages 259–284. Springer, 2014.
- Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, and Jianzhu Ma. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. In *Advances in Neural Information Processing Systems*.
- Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. Survae flows: Surjections to bridge the gap between vaes and flows, 2020. URL <https://arxiv.org/abs/2007.02731>.
- George Papamakarios, Eric T Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22(57):1–64, 2021.
- Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2015. URL <https://arxiv.org/abs/1505.05770>.
- Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL <https://proceedings.mlr.press/v37/sohl-dickstein15.html>.

- Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL <https://proceedings.neurips.cc/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf>.
- Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=PXTIG12RRHS>.
- Vincent Stimper, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Resampling base distributions of normalizing flows. 2021. doi: 10.48550/ARXIV.2110.15828. URL <https://arxiv.org/abs/2110.15828>.
- Esteban G Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.
- Lilian Weng. What are diffusion models? *lilianweng.github.io*, Jul 2021. URL <https://lilianweng.github.io/posts/2021-07-11-diffusion-models/>.
- Hao Wu, Jonas Köhler, and Frank Noé. Stochastic normalizing flows, 2020. URL <https://arxiv.org/abs/2002.06707>.
- Qinsheng Zhang and Yongxin Chen. Diffusion normalizing flow. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 16280–16291. Curran Associates, Inc., 2021. URL <https://proceedings.neurips.cc/paper/2021/file/876f1f9954de0aa402d91bb988d12cd4-Paper.pdf>.