DENSITY APPROXIMATION IN DEEP GENERATIVE MODELS WITH KERNEL TRANSFER OPERATORS

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models which use explicit density modeling (e.g., variational autoencoders, flow-based generative models) often involve finding the optimal mapping (i.e., transfer operator) from a known distribution, e.g. Gaussian, to the input (unknown) distribution. This often requires searching over a class of non-linear functions (e.g. functions that can be represented by a deep neural network). While effective in practice, the associated computational/memory costs can increase rapidly, usually as a function of the performance that is desired in an application. We propose a substantially cheaper (and simpler) distribution matching strategy by leveraging recent developments in neural kernels together with ideas from known results on kernel transfer operators. We show that our formulation enables highly efficient distribution approximation and sampling, and offers empirical performance that compares very favorably with powerful baselines, but with significant savings in runtime. We show that the algorithm also performs well in the small sample size settings.

1 INTRODUCTION

Given *i.i.d.* samples of the data X with an unknown density p_X, most generative modeling approaches seek to estimate or derive a parametric density function p_θ that closely resembles properties of p_X. Ideally, one hopes that $p_\theta = p_X$ and therefore samples drawn from p_θ will closely resemble the training data X. In modern deep generative models, one often approaches this question by utilizing a *latent space*. That is, we assume that there is some latent variable Z associated with the observed data X that follows a *known* distribution p_Z (e.g., isotropic Gaussian).

Consider generative models with an explicit decoder or generator structure, for instance, generative adversarial networks (GANs) Goodfellow et al. (2014) or variational autoencoders (VAEs) Kingma & Welling (2013). The parameterized empirical density $p_\tau(X)$ can be written as $\int p_\tau(X|Z)p(Z)dz$ where Z is the latent variable with a suitable prior and the conditional density $p_\tau(X|Z)$ is usually modeled with a multi-layer perceptron or a convolutional neural network. Such a decoder shapes (or non-linearly transforms) a simple prior distribution p_z into a complex p_X. Training a generative model of this form will involves a numerical optimization over an appropriate loss function, e.g., associated with the likelihood Kingma & Welling (2013).

In both examples above, the goal is to find an appropriate parameter τ such that p_τ marginalized over Z satisfies $p_\tau \approx p_X$. At a high level, given the data variable $X \in \mathcal{X}$ distributed according to some unknown distribution p_X, a key piece of the workflow in deep generative models is to learn a mapping (or transformation) or a *forward operator* as defined below.

Definition 1.1 (Forward operator)**.** *A forward operator $f^* \in \mathcal{C} : \mathcal{Z} \to \mathcal{X}$ is defined to be a mapping associated with some latent variable $Z \in \mathcal{Z} \sim p_Z$ such that $f^* = arg\ min_{f\in\mathcal{C}} d(p_{f(Z)}, p_X)$ for some function class $\mathcal{C}$ and distance measure d.*

The **starting point** of our work is to evaluate the extent to which we can radically simplify the forward operator for density approximation in deep generative models – based on directly using or re-purposing existing results in a manner that we still obtain satisfactory empirical performance. Consider an axiomatic description of a forward operator (as defined in (1.1) based on the following reasonable properties: **(a)** Upon convergence, the learned operator f minimizes the distance/divergence between p_X and $p_{f(Z)}$ over all possible operators of a certain class. **(b)** The training procedure directly learns

the mapping from the prior distribution, rather than an approximation. **(c)** The forward operator f is efficient both for training and inference.

It would appear that the above list of criteria violates the "no free lunch rule", and some compromise must be involved. Our goal is to investigate precisely this trade-off and the design choices that are needed to make it work. Specifically, a well studied result in dynamical systems, namely the existence of a Perron-Frobenius operator Lemmens & Nussbaum (2012), suggests an alternative *linear* route to model the forward operator. As we will discuss shortly, we show that bringing together recent results in neural kernels and kernel transfer operators, the forward operator in generative models can be efficiently approximated as the estimation of a **closed-form linear operator** in the reproducing kernel Hilbert space (RKHS).

Contributions. Our results are largely based on the existing literature in kernel methods and dynamical systems, but we demonstrate their relevance in generative modeling and complement recent results that emphasize the links between deep generative models and dynamical systems. Our main contributions are **(i)** We propose a simple framework for transferring a known prior density *linearly* to an unknown data density in RKHS, which is equivalent to learning a nonlinear forward operator in the input space. **(ii)** We empirically evaluate this idea in multiple density approximation scenarios and show very competitive performance for image generation tasks on popular datasets including CelebA and also show its use for generative modeling tasks in a small sample size setting.

Notations used			Meaning
Z	X	$\triangleq$	Random variable
$\mathcal{Z}$	$\mathcal{X}$	$\triangleq$	Domain
p_Z	p_X	$\triangleq$	Density function
k	l	$\triangleq$	Kernel function
$\mathcal{H}$	$\mathcal{G}$	$\triangleq$	RKHS
$\phi(z) \quad k(z,\cdot)$	$\psi(x) \quad l(x,\cdot)$	$\triangleq$	feature map
$\mathcal{E}_k$	$\mathcal{E}_l$	$\triangleq$	Mean embedding operator
$\mu_Z \quad \mathcal{E}_k p_Z$	$\mu_X \quad \mathcal{E}_l p_X$	$\triangleq$	Kernel mean embedding

Table 1: Commonly used notations in this paper.

2 Simplifying forward operator estimation

Forward operator as a dynamical system: Viewing the forward operator f^* as a (deterministic) continuous dynamical systems is similar to how flow-based generative models are considered explicitly as a continuous dynamical system in Neural ODE Chen et al. (2018). As a simple illustration, the dynamics on an element $z(t_0) \in \mathcal{X}$ by applying f^* is governed by the system $x = z(t_0) + \int_{t_0}^{t_1} \Delta_t(z(t))dt$. Integrating the RHS by $g^*(z(t_0)) = \int_{t_0}^{t_1} \Delta_t(z(t))dt$. The forward operator $f^* = (I + g^*) : \mathcal{X} \to \mathcal{X}$, therefore, controls the discrete evolution of $z(t_0)$ at time t_1 as $x = f^*(z(t_0))$. The marginal density over a subset of the state space $\Lambda \subseteq \mathcal{X}$ can therefore be expressed as $\int_\Lambda p_X(x)dx = \int_{z \in f^{*-1}(\Lambda)} p_Z(z)dz$.

Towards a one-step estimation of forward operator: Both f^* and f^{*-1} are usually highly nonlinear functions. Learning f^* in general requires searching in a large space of nonlinear functions (e.g., possible functions represented by a given neural network architecture) and sometimes evaluations of the RHS integral or its lower bound (i.e., ELBO in VAE and exact likelihood in flow), which can both be expensive in practice. Nevertheless, the dynamical systems literature suggests a linear extension of the transformation of p_Z, namely the *Perron-Frobenius* operator or transfer operator, that conveniently transfers p_Z to p_X.

Definition 2.1 (Perron-Frobenius operator Mayer (1980))**.** *The Perron-Frobenius (PF) operator $\mathcal{P} : L^1(\mathcal{X}) \to L^1(\mathcal{X})$ is an infinite-dimensional linear operator defined as $\int_\Lambda (\mathcal{P}p_Z)(x)dx = \int_{z \in f^{-1}(\Lambda)} p_Z(z)dz$ for all $\Lambda \subseteq \mathcal{X}$.*

With this definition of the PF operator, $\mathcal{P}$, we have $p_X = \mathcal{P}p_Z$. If we assume that such an operator can be efficiently estimated, we can use it to transfer the tractable probability density p_Z to the target density p_X. However, since $\mathcal{P}$ is an infinite-dimensional operator on $L^1(\mathcal{X})$, it is impractical to instantiate it explicitly and exactly.

Kernel embedded form of the PF operator: A natural extension or variation that has been explored Klus et al. (2020) is to represent $\mathcal{P}$ by an infinite set of functions, say by mapping the given distribution

into an infinite dimensional space of functions through the *kernel trick*. There, for a characteristic kernel l, the *kernel mean embedding* uniquely identifies an element $\mu_X \in \mathcal{G}$ for $p_X \in L^1(\mathcal{X})$. Therefore, solving the dynamics of p_Z in its embedded form allows approximating $\mathcal{P}$ with an infinite basis. Recent literatures suggest such a linear operator in RKHS that defines the dynamics between two embedded densities.

Definition 2.2 (Embedded Perron-Frobenius operator Klus et al. (2020))**.** *Given $p_X \in L^1(\mathcal{X})$ and $p_Z \in L^1(\mathcal{X})$. Let $\mu_X = \mathcal{E}_l p_X$ and $\mu_Z = \mathcal{E}_k p_Z$ be their corresponding mean kernel embeddings. The kernel Perron-Frobenius (kPF) operator, denoted by $\mathcal{P}_\mathcal{E} : \mathcal{H} \to \mathcal{G}$, is defined as $\mu_X = \mathcal{P}_\mathcal{E} \mu_Z = C_{XZ} C_{ZZ}^{-1} \mu_Z$ under the following conditions:* ***(i)*** *C_{ZZ} is injective* ***(ii)*** *$\mu_t \in$ Range(C_{ZZ})* ***(iii)*** *$E[g(X)|Z = \cdot] \in \mathcal{H}$ for any $g \in G$.*

Transferring embedded densities with the PF operator: The embedded PF operator is a powerful tool that allows transferring the embedded densities in RKHS. Further, the commutativity between the (embedded) PF operator and the mean embedding operator is given in Klus et al. (2020), showing the equivalence of $\mathcal{P}_\mathcal{E}$ to $\mathcal{P}$ when l is characteristic.

In practice with finite data, $\{x_i\}_{i\in[n]} \sim X^n$ and $\{z_i\}_{i\in[n]} \sim Z^n$, $\mathcal{P}_\mathcal{E}$ is given by its empirical estimate $\hat{\mathcal{P}}_\mathcal{E} = \hat{C}_{XZ}\hat{C}_{ZZ}^{-1} = \Psi(\Phi^T\Phi + \lambda n I)^{-1}\Phi^T$ where $\Phi = [k(z_1, \cdot), \cdots, k(z_n, \cdot)], \Psi = [l(x_1, \cdot), \cdots, l(x_n, \cdot)]$ are the feature matrices corresponding to samples of X and Z. Error analysis of this estimate is described in Klus et al. (2020).

Mapping Z to $\mathcal{G}$: Let us define $\phi(z) = k(z, \cdot)$ and $\psi(x) = l(x, \cdot)$ as the feature maps of kernels k and l. We can rewrite μ_X as

$$\mu_X = \mathcal{P}_\mathcal{E}\mathcal{E}_k p_Z = \mathcal{P}_\mathcal{E} E_Z[\phi(Z)] = E_Z[\mathcal{P}_\mathcal{E}(\phi(Z))] = E_Z[l\left(\psi^\dagger\left(\mathcal{P}_\mathcal{E} k\left(Z, \cdot\right)\right), \cdot\right)] \ (1).$$

Here $\psi^\dagger$ is the inverse, or *preimage map*, of ψ. Such an inverse, in general, may not exist. In most cases, an approximate preimage is considered instead Kwok & Tsang (2004); Honeine & Richard (2011); Pandey et al. (2019). In what follows, we will temporarily assume that an exact preimage map exists and is tractable to compute.

Let us define $\Psi^* = \hat{\mathcal{P}}_\mathcal{E} k(Z, \cdot)$ as the *transferred sample* in $\mathcal{G}$ using the empirical embedded PF operator $\hat{\mathcal{P}}_\mathcal{E}$. Then in the following proposition, we show that asymptotically the preimages of the transferred samples converge to the target distribution.

Proposition 2.1. *As $n \to \infty$, $\psi^\dagger\left(\Psi^*\right) \xrightarrow{d} p_X$. In other words, the preimage of the transferred sample approximately conforms p_X under the previous assumptions when n is large.*

The above statement can be shown by noticing that (1) has the same form as the kernel mean embedding $\mu_X = \mathcal{E}_l p_X = E_X[l(X, \cdot)]$.

With all building blocks in hand, we now present an algorithm for sample generation using the PF operator. A detailed description is presented in Alg. 1. The idea is simple yet powerful: at training time, we first construct the empirical embedded PF operator using the training data $\{x_i\}_{i\in[s]}$ and samples of the known prior $\{z_i\}_{1\in[n]}$. At inference time, we will use the constructed operator to transfer new points sampled from the known distribution to the target feature map, and construct their preimages as the generated output samples.

Algorithm 1 Sample Generation from embedded PF Operator

1: **Input:** Training data $\{x_i\}_{i\in[n]}$, $\{z_i\}_{i\in[n]}$; Sample z^* from p_Z
2: Compute

$$\mathbf{s} = \langle \Psi, \hat{\mathcal{P}}_\mathcal{E} k(z^*, .)\rangle = \Psi^T\Psi(\Phi^T\Phi + \lambda n I)^{-1}\Phi k(z^*, .)$$
$$= L\left(K + \lambda n I\right)^{-1}\mathbf{v},$$

where $\mathbf{v} = [k(z_i, z^*)]_i \in \mathbf{R}^n$ and

$$L = \left[l(x_i, x_j)\right]_{i,j}, K = \left[k\left(z_i, z_j\right)\right]_{i,j} \in \mathbf{R}^{n\times n}$$

3: Select $\{\mathbf{x}_j\}_{j=1}^{\gamma}$ as the training samples corresponding to the top $\gamma > 0$ values in $\mathbf{s}$
4: $x^* = \text{preimage}\left(\{\mathbf{x}_j\}_{j=1}^{\gamma}, \mathbf{s}\right)$.
5: **Output** generated sample x^*

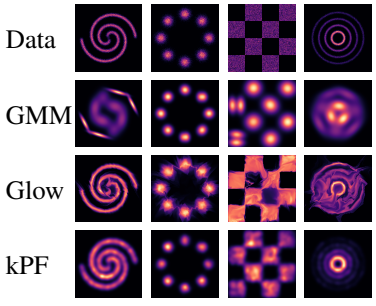

Figure 1: Estimated densities

	Glow‡	CAGlow‡	Vanilla VAE	WAE†	2-stage VAE	SRAE$_{Glow}$	SRAE$_{GMM}$	SRAE$_{RBF-kPF}$ (ours)	SRAE$_{NTK-kPF}$ (ours)
MNIST	25.8	26.3	**13.7**	20.4	18.3	23.7	16.7	21.7	21.5
CIFAR-10	-	-	111.0	117.4	110.3	110.7	79.2	**77.9**	**77.5**
CelebA	103.7	104.9	52.1	53.7	44.7	59.8	42.0	**41.9**	**41.0**

Table 2: Comparative FID values. SRAE indicates an autoencoder with hyperspherical latent space and spectral regularization following Ghosh et al. (2020). Subscripts indicates the corresponding sampling techniques on latent space. For other state-of-the-art non-adversarial generative models, we compared with VAE, Two-stage VAE, Wasserstein AE, and Glow variants. Results reported from ‡: Liu et al. (2019). †: Ghosh et al. (2020).

3 Experimental Results

Goals. For our experiments, we mainly seek to answer two questions: dxv**(a)** With sufficient data, can the proposed method generate new data with comparable performance with other state-of-the-art generative models? **(b)** If only limited data samples were given, can the proposed method still estimate the density with reasonable accuracy?

Datasets/setup. To answer the first question, we evaluate our method on CelebA, where the number of data samples is sufficient. To generate images, we use a pretained regularized autoencoder Ghosh et al. (2020) with a latent space restricted to the hypersphere (denoted by SRAE) to encourage *smooth* latent representations, and construct the corresponding PF operator using samples of a simple prior (i.e. isotropic Gaussian) and the latent representations of the training data. To generate new samples, prior samples are transferred by the PF operator, and their approximate preimages were decoded using the AE decoder. We compare our results with other state-of-the-art VAE variants (Two-stage VAE Dai & Wipf (2019), WAE Arjovsky et al. (2017)), and flow-based generative models (Glow Kingma & Dhariwal (2018), CAGlow Liu et al. (2019)). The second question is motivated by Arora et al. (2020), where kernel methods consistently outperform neural networks in small data settings. In order to evaluate this more challenging case, we randomly pick 100 training samples ($< 1\%$ of the full dataset) from CelebA and evaluate the FIDs for all density approximators on the latent space.

Two-stage VAE SRAE$_{GMM}$ SRAE$_{Glow}$ SRAE$_{NTK\text{-}kPF}$

Figure 2: Comparison of different sampling techniques on latent space of AE trained on CelebA 64x64. *Left to right: (1) samples of Two-stage VAE (2) samples of SRAE+GMM (3) samples of SRAE+Glow (4) samples of SRAE+NTK-kPF using 10k latent points.*

Results. We evaluate the quality by calculating the Fréchet Inception Distance (FID) Heusel et al. (2017) with 10K generated images from each model. All implemented models shares the same encoder/decoder structure used in Ghosh et al. (2020). Subscript indicates different types of density estimators learned on the latent variables, including Glow Kingma & Dhariwal (2018), Gaussian mixture model (GMM), and two proposed PF operators with Gaussian kernel as the input kernel (RBF-kPF) and with NTK as the input kernel (NTK-kPF). We further note that, since the estimated PF operator can be computed in closed form, it incurs an over $50\times$ reduction in learning time compared with neural network based approaches (i.e. VAE and Glow).

Comparative results using training on the entire data is shown in Table 2, with some generated samples in Fig. 2. Furthermore, with only 100 training samples, our models obtain FID values of 40.6 (RBF-kPF) and 40.9 (NTK-kPF) compared with 59.3 (VAE), 77.0 (Glow) and 39.6 (GMM). This clearly dictates the effectiveness of the proposed method in the limited data setting.

4 Conclusions

In this paper, we show that with the help of recent developments in regularized autoencoders and neural kernels, a linear kernel transfer operator can potentially be an efficient substitute for the forward operator in some generative models. Our proposed method shows comparable empirical results to other state-of-the-art generative models on CelebA, while enjoying much better computational efficiency. Furthermore, we showed performance gain using our proposed method even in case of few number of samples presented during training.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Sanjeev Arora, Simon S. Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu. Harnessing the power of infinitely wide deep nets on small-data tasks. In *International Conference on Learning Representations*, 2020.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 6571–6583. Curran Associates, Inc., 2018.

Bin Dai and David Wipf. Diagnosing and enhancing VAE models. In *International Conference on Learning Representations*, 2019.

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

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27, pp. 2672–2680. Curran Associates, Inc., 2014.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 6626–6637. Curran Associates, Inc., 2017.

Paul Honeine and Cedric Richard. Preimage problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, 28(2):77–88, 2011.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 10215–10224. Curran Associates, Inc., 2018.

Stefan Klus, Ingmar Schuster, and Krikamol Muandet. Eigendecompositions of transfer operators in reproducing kernel hilbert spaces. *Journal of Nonlinear Science*, 30(1):283–315, 2020.

JT-Y Kwok and IW-H Tsang. The pre-image problem in kernel methods. *IEEE transactions on neural networks*, 15(6):1517–1525, 2004.

Bas Lemmens and Roger Nussbaum. *Nonlinear Perron-Frobenius Theory*, volume 189. Cambridge University Press, 2012.

Rui Liu, Yu Liu, Xinyu Gong, Xiaogang Wang, and Hongsheng Li. Conditional adversarial generative flow for controllable image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Dieter H Mayer. The ruelle-araki transfer operator in classical statistical mechanics. 1980.

Arun Pandey, Joachim Schreurs, and Johan AK Suykens. Generative restricted kernel machines. *arXiv preprint arXiv:1906.08144*, 2019.