Information Theoretic Structured Generative Modeling

Bo Hu\textsuperscript{1}, Shujian Yu\textsuperscript{2, 3}, José C. Príncipe\textsuperscript{1}

\textsuperscript{1}University of Florida, Gainesville, FL 32611, USA
hubo@ufl.edu; principe@cnl.ufl.edu

\textsuperscript{2}UiT - The Arctic University of Norway, 9037 Tromsø, Norway

\textsuperscript{3}Xi’an Jiaotong University, Xi’an 710049, Shaanxi, China
yusj9011@gmail.com

Abstract
Rényi’s information provides a theoretical foundation for tractable and data-efficient non-parametric density estimation, based on pair-wise evaluations in a reproducing kernel Hilbert space (RKHS). This paper extends this framework to parametric probabilistic modeling, motivated by the fact that Rényi’s information can be estimated in closed-form for Gaussian mixtures. Based on this special connection, a novel generative model framework called the structured generative model (SGM) is proposed that makes straightforward optimization possible, because costs are scale-invariant, avoiding high gradient variance while imposing less restrictions on absolute continuity, which is a huge advantage in parametric information theoretic optimization. The implementation employs a single neural network driven by an orthonormal input appended to a single white noise source adapted to learn an infinite Gaussian mixture model (ImoG), which provides an empirically tractable model distribution in low dimensions. To train SGM, we provide three novel variational cost functions, based on Rényi’s second-order entropy and divergence, to implement minimization of cross-entropy, minimization of variational representations of $\phi$-divergence, and maximization of the evidence lower bound (conditional probability). We test the framework for estimation of mutual information and compare the results with the mutual information neural estimation (MINE), for density estimation, for conditional probability estimation in Markov models as well as for training adversarial networks. Our preliminary results show that SGM significantly improves MINE estimation in terms of data efficiency and variance, conventional and variational Gaussian mixture models, as well as the performance of generative adversarial networks.

Introduction
The traditional information descriptors in probability spaces are Shannon’s entropy, Mutual Information, and the Kullback–Leibler divergence (relative entropy). Since the probability density function of the data in machine learning is often unknown, applying these descriptors requires well-behaved estimators or approximations. The objective of information theoretic learning (ITL) is to provide a clear and theoretically supported view of probabilistic modeling algorithms. There are basically two different avenues in the literature: non-parametric estimators and parametric estimators. Rényi’s second-order entropy and divergence have been broadly used for designing non-parametric estimators following the principle of sample-based pair-wise evaluations in reproducing kernel Hilbert space (RKHS). The main difficulty of this non-parametric framework is its dependence on a free parameter that yields a bias. If the bias can be tolerated (e.g. optimization of ITL cost functions), its appeal is interpretability, data efficiency, and good performance.

The other avenue optimizes a universal parametric model to estimate the ITL descriptors, which usually follows three types of principles: minimizing cross-entropy, minimizing variational representations of $\phi$-divergence, and maximizing the evidence lower bound (ELBO). For example, minimization of cross-entropy, which is simple to implement from the data distributions, yields a model that is an upper bound for the data entropy. Solving variational representations of $\phi$-divergence based on a function of the probability ratio $p(x)/q(x)$ quantifies the divergence between two distributions as an upper bound of the model mapping. Recently, generative adversarial networks (GANs) take this approach with one network approximating $\phi$-divergence and another network producing an intractable model distribution to minimize the approximated $\phi$-divergence. In variational inference, if direct optimization over cross-entropy is difficult, ELBO introduces a latent variable with conditional probabilities to optimize the parameters of an underlying model that has a simpler structure.

However, experience has shown that optimizing all these costs is a challenging task. The main reason is that Shannon’s cross-entropy or Kullback–Leibler divergence require absolute continuity of two Lebesgue measures $Q \gg P$, which is highly restrictive during adaptation of any parameterized model that yields the distribution. Without this condition the bounds do not hold. Secondly, in order to avoid estimating the nonlinear $\log q(x)$, the asymptotic equipartition property (AEP) is normally utilized, which is not data efficient. Thirdly, the ultimate task is to produce a valid probability distribution, which requires constrained optimization methodologies. All three aspects also create high variance in the gradient and forbid the application of simple optimization techniques. The field conventionally uses expectation maximization (EM), stochastic approximation, or what is also called “reparameterization tricks”.
The main contribution of this paper is to propose a unified new set of information theoretic cost functions based on the theory of Rényi’s entropy and divergence \[17\] that improves these three issues. Rényi’s entropy and the corresponding divergence are related to the special case of f-divergence when the convex function f is taken to be a polynomial. We derive three variational forms for Rényi’s second-order entropy, divergence, and conditional entropy that corresponds to the three mentioned principles. The important property of Rényi’s information as opposed to Shannon is that the log is outside of the integral, which simplifies the estimation \[11\] and provides a closed-form solution for mixture of Gaussian \[13\], which dramatically simplifies the computation for the versatile class of generalized Gaussian mixture models.

The second contribution is to propose a new generative model known that Gaussian mixture models maximizes the log

Cross-entropy and GMM

random variables.

and divergence. Throughout the paper, we assume that the proposed framework, the discriminator network will produce probabilities, and we also show an example of the newly proposed estimators, we show its versatility by estimating conditional distributions or conditional distributions by approximating local sample densities by their mean and variance, assuming a versatile generalized Gaussian mixture model.

SGM uses a single neural network to produce an infinite mixture of Gaussian (IMoG) that can be trained efficiently assuming a versatile generalized Gaussian mixture model. IMoG approximate pair-wise evaluations and the theory of RKHS \[1\]. This most obvious downside is that SGM uses a single neural network to produce an intractable model distribution for this upper bound to hold is very restrictive. Suppose there exists x such that \(p(x) > 0\) and \(g_0(x) = 0\), it is easy to show that \(\mathrm{CE}(p, g_0) \to \infty\). On the other hand, the responsibility \(g_0(x, z)/g_0(x)\) is taken to be finite such that the RHS has finite values. In this case, this bound no longer exists.

\[f\text{-divergence and GAN:}\] Another important branch of costs is the f-divergence. Given a convex function f with \(f(1) = 0\), f-divergence has the form \(D_f(p||q) = \int_X q(x)f(p(x)/q(x))d\mu\) that evaluates a “distance” between two distributions. A variational form of \(D_f(p||q)\) takes the convex conjugate of \(f(21)\), which becomes \[13\]:

\[D_f(p||q) = \sup_{T \in T} \left( \mathbb{E}_{p}[T(x)] - \mathbb{E}_q[f^*(T(x))] \right), \tag{3}\]

where T is an arbitrary class of functions \(T: \mathcal{X} \to \mathbb{R}\). The bound is tight when \(T_0(x) = f'(p(x)/q(x))\). Although \(T_0\) is written in the form of \(p(x)/q(x)\), the absolute continuity condition also depends on the choice of f.

Any universal parametric model such as kernel functions \[6, 7\] or neural networks \[22\] can be used to approximate \(T^{\prime}\) by maximizing the lower bound. GAN uses a discriminator network to approximate \(T\), and a generator network to produce an intractable model distribution \(g_0\) that minimizes the approximated f-divergence \[12, 13\]. The most obvious downside is that \(g_0\) is intractable. Secondly, although \(T_0\) should approach \(T_0(x)\) related to the probability ratio, there is no guarantee in GAN.

\[\text{Rényi’s entropy and divergence:}\] We also mention Rényi’s entropy and divergence of order \(\alpha\)

\[H_\alpha(p) = \frac{1}{1-\alpha} \log \int_X p^\alpha(x)d\mu, \tag{4}\]

\[D_\alpha(p||q) = \frac{1}{1-\alpha} \log \int_X p^\alpha(x)q^{1-\alpha}(x)d\mu.\]

Rényi’s divergence is related to the special case of \(H_\alpha\) when f is chosen to be a polynomial. The existence of \(D_\alpha(p||q)\) requires \(Q \gg P\) for all \(\alpha \geq 1\). When \(\alpha = 2\), Rényi’s second-order divergence matches the \(\chi^2\)-divergence. Rényi’s second-order entropy and divergence are the foundations of non-parametric density estimators based on sample-based pair-wise evaluations and the theory of RKHS \[1\].

This paper further expands this idea.

\[\text{Background}\]

We start with an introduction of cross-entropy, f-divergence, and a short discussion about Gaussian mixture models (GMM) and generative adversarial networks (GAN) from the perspective of ITL. Then we introduce Rényi’s entropy and divergence. Throughout the paper, we assume that the density functions exist and are Lebesgue measurable. We use the definition of differential entropy for the continuous random variables.

Cross-entropy and GMM: Let a density function \(q(x)\) be given. Taking the expectation of \(-\log q(x)\) over \(p(x)\) yields the cross-entropy

\[\mathrm{CE}(p, q) = \int_X p(x) \log p(x) + \mathrm{KL}(p||q) \geq -\int_X p(x) \log p(x) = H(p). \tag{1}\]

Thus, minimizing \(\mathrm{CE}(p, q)\) yields \(H(p)\), and the tightness of the bounds depends directly on \(\mathrm{KL}(p||q)\). It is well-known that Gaussian mixture models maximizes the log likelihood of data, but it can also be formulated as minimizing the cross-entropy. Let the model density be \(g_0(x) = \sum_{i=1}^N w_i \mathcal{N}(x - m_i, A_i)\). By \(1\), the cross-entropy satisfies

\[\mathrm{CE}(p, g_0) = -\int_X p(x) \log \sum_{i=1}^N w_i \mathcal{N}(x - m_i, A_i)d\mu \geq H(p). \]

We write \(g_0(x, z = i) = w_i \mathcal{N}(x - m_i, A_i)\). Since \(\partial \mathrm{CE}(p, g_0)/\partial \theta = 0\) does not have a closed-form solution, expectation maximization (EM) and variational inference optimize an upper bound of \(\mathrm{CE}(p, g_0)\):

\[\mathrm{CE}(p, g_0) \leq \mathrm{CE}\left(p(x) \frac{g_0(x, z)}{g_0(x)}, g_0(x, z)\right), \tag{2}\]

Ideally, the upper bound is tight when \(p(x)g_0(x, z)/g_0(x) = g_0(x, z)\), i.e., \(g_0(x) = p(x)\), which yields the same solution as minimizing \(\mathrm{CE}(p, g_0)\). EM iteratively updates \(g_{\text{new}} \rightarrow \max_{\theta} \mathrm{CE}(p(x)g_0(x, z)/g_0(x, z), g_0(x, z))\). Gradient methods can also be used \[5\] \[20\]. However, the condition for this upper bound to hold is very restrictive. Suppose there exists \(x\) such that \(p(x) > 0\) and \(g_0(x) = 0\), it is easy to show that \(\mathrm{CE}(p, g_0) \to \infty\). On the other hand, the responsibility \(g_0(x, z)/g_0(x)\) is taken to be finite such that the RHS has finite values. In this case, this bound no longer exists.
Properties of MoG: Finally, we mention of pair-wise properties of mixture of Gaussian (MoG) that we utilize for the design. Given any two Gaussian density functions, it can be shown that \( \int X N(x - m_i, A_i)N(x - m_j, A_j)dmu = N(m_i - m_j, A_i + A_j) \). Now suppose we have a MoG with a density function \( T_\varphi(x) = \sum_{i=1}^{N} w_i N(x - m_i, A_i) \). It follows that \( T_\varphi(x) \) satisfies
\[
\int X T_\varphi^2(x)dmu = \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j N(m_i - m_j, A_i + A_j),
\]
which is fully determined by \( \varphi \) with a closed-form solution based on pair-wise relations \[18\].

We can further show that the same conclusion holds for an infinite mixture of Gaussian (IMoG) in a Bayesian setting, where \( \varphi \) is determined by an arbitrary distribution \( P_\varphi \). Thus the model density has the form \( T_\varphi(x) = \mathbb{E}_{\varphi \sim P_\varphi} [w N(x - m, A)] \). Similarly it can be shown that
\[
\int X T_\varphi^2(x)dmu = \mathbb{E}_{\varphi \sim P_\varphi} [w_1 w_2 N(m_1 - m_2, A_1 + A_2)].
\]
In the following sections, we show how we use a neural network to produce the model density \( T_\varphi(x) \). While imposing a MoG can introduce a bias \[5\], the advantage of controlling the variance with a model is very appealing and surpasses the former.

Proposed Cost Functions
In this section, we introduce two new variational cost functions \( J_p(T) \) and \( J_{p,q}(T) \), which accepts upper bound related to Rényi’s second-order entropy \( H_2(P) \) and second-order divergence \( D_2(P\|Q) \).

Variational Rényi’s second-order entropy: Similar to the cross-entropy, we introduce a functional \( J_p(T) \) with the form
\[
J_p(T) = \frac{\mathbb{E}_p[T(x)]}{\langle \int X T^2(x)dmu \rangle^{\frac{1}{2}}},
\]
over the set \( T \) that contains all non-negative measurable functions with \( \int X T^2(x)dmu > 0 \). For simplicity, we write the inner product \( \int X p(x)T(x)dmu := \langle p, T \rangle \), the norm \( \int X p^2(x)dmu = \langle p, p \rangle \), and \( \int X T^2(x)dmu = \langle T, T \rangle \). By Cauchy-Schwarz inequality, we have \( \langle p, T \rangle \leq \sqrt{\langle p, p \rangle \langle T, T \rangle} \). It follows that \( J_p(T) = \langle p, T \rangle / \sqrt{\langle T, T \rangle} \leq \langle p, p \rangle \). Therefore we obtained
\[
\sup_{T \in T} J_p(T) = \left( \int X p^2(x)dmu \right)^{\frac{1}{2}}.
\]
There are two important conclusions. First, the upper bound only depends the data. Second, the solution is \( T_0(x) = \beta p(x) \) with \( \beta \) taken to be any arbitrary positive scalar. The function \( T \) can be approximated by any parametric model.

Rényi’s second-order divergence: Following a similar idea, we introduce a functional \( J_{p,q}(T) \) for Rényi’s second-order divergence, written as
\[
J_{p,q}(T) = \frac{\mathbb{E}_p[T(x)]}{\langle \mathbb{E}_q[T^2(x)] \rangle^{\frac{1}{2}}},
\]
Similarly, by applying Cauchy-Schwartz inequality,
\[
J_{p,q}(T) = \frac{\langle p(x) \sqrt{q(x)} \rangle}{\langle T \sqrt{q(x)} \rangle \langle T, T \rangle^{\frac{1}{2}}} \leq \frac{\langle p(x), p(x) \rangle^{\frac{1}{2}}}{\langle T \sqrt{q(x)} \rangle}.\]
Therefore we proved that there is an upper bound for \( J_{p,q}(T) \)
\[
\sup_{T \in T} J_{p,q}(T) = \left( \int X p^2(x)dmu / q(x) \right)^{\frac{1}{2}}.
\]
The supremum is attained as \( T_0(x) = \beta p(x) / q(x) \), which is a scaled value of the probability ratio. Combining with \[4\], it can be shown that the upper bounds of \( J_p(T) \) and \( J_{p,q}(T) \) are related to Rényi’s second-order entropy and divergence as \( H_2(P) = -2 \log \sup_{T \in T} J_p(T) \) and \( D_2(P\|Q) = -2 \log \sup_{T \in T} J_{p,q}(T) \).

Proposed Algorithms
Now we show how \( J_p(T) \) and \( J_{p,q}(T) \) can be employed for model training. First, optimizing \( J_{p,q}(T) \) shares the same procedure as optimizing any variational form of f-divergence \[6, 7, 22\]. It can also train a generative adversarial network with the discriminator network producing the probability ratio. Optimizing \( J_p(T) \) is challenging because the normalizing term \( \int X T^2(x)dmu \). One unwise choice is to impose a uniform distribution such that \( \int X T^2(x)dmu = \mathbb{E}_u[T^2(x)]/|Z| \). This is not practical since the support of the data distribution may be unknown. Therefore, we use the properties of MoG introduced in the Background section that if \( T(x) \) is a mixture of Gaussian (MoG) or an infinite mixture of Gaussian (IMoG), the term \( \int X T^2(x)dmu \) will have a closed-form solution \[9\]. In this section, we show how we use a neural network to produce this density function of IMoG. We first propose the following density approximation algorithm based on this property.

Density approximation: Similar to GAN, we impose two types of network inputs \( z \) and \( c \). We first define the discrete orthogonal vectors \( z \). We define a vector \( z_i = [z_i(1), z_i(2), \ldots, z_i(N)]^T \), with each element satisfying \( z_i(j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \). Now we obtained a orthonormal set of \( N \) one-hot vectors \( \{z_1, z_2, \ldots, z_N\} \). As for \( c \), we assume it is sampled from a uniform distribution \( c \sim U(0, 1) \). We call the combination of the orthonormal set and the uniformly distributed noise the scanning vectors, whose role is to span the sample space to quantify local structures. The scanning vectors form the inputs to the neural network.

The output of the neural net defines \( F_\varphi \). With \( z_i \) and \( c \) as the input, we denote the output of the neural net as \( \{f^{(w)}_\theta(z_i, c), f^{(m)}_\theta(z_i, c), f^{(A)}_\theta(z_i, c)\} \), which defines the model density function
\[
g_\theta(x) = \mathbb{E}_c \left[ \sum_{i=1}^{N} f^{(w)}_\theta(z_i, c)N(x - f^{(m)}_\theta(z_i, c), f^{(A)}_\theta(z_i, c)) \right].
\]
(12)
Rewriting $w_i(c) = f^u_\theta(z_i, c)$, $m_i(c) = f^m_\theta(z_i, c)$, and $A_i(c) = f^{(A)}_\theta(z_i, c)$, we have
\[ N_{i,j} = \mathcal{N}(m_i(c_1) - m_j(c_2), A_i(c_1) + A_j(c_2)) \]
\[ \int g_\theta^2(x) d\mu = E_{c_1,c_2} \left[ \frac{1}{N^2} \sum_{i=1}^N w_i(c_1) w_j(c_2) N_{i,j} \right], \tag{13} \]
which is fully defined by $\mu$, irrelevant to the data distribution and avoids an empirical estimation over $X$. The optimization problem can be written as
\[ \text{maximize}_{g_\theta} J_{P_{0}}(g_\theta) = \frac{\mathbb{E}_x [g_\theta^2(x) \mu(x)]}{\left( \int g_\theta^2(x) d\mu \right)^{\frac{1}{2}}}. \tag{14} \]

The added advantage is that the model can be trained efficiently by gradient ascent. The upper bound of the cost function is given by (9) and the bound is tight if $\mu(x) = p(x)$. We call this approach the structured generative model (SGM).

One difficulty for optimizing directly with such a cost is the variance of the gradient, which can be reduced in a procedure introduced in (23). We define $k_\theta = \mathbb{E}_p [g_\theta(x)]$ and $v_\theta = \int g_\theta^2(x) d\mu$, the gradient of $J_{P_{0}}(g_\theta)$ has the form $\frac{\partial J_{P_{0}}(g_\theta)}{\partial \theta} = \left( \frac{\partial k_\theta}{\partial \theta} \right) / \sqrt{v_\theta} - \frac{1}{2} \left( \frac{k_\theta | \partial v_\theta}{\partial \theta} / \sqrt{v_\theta} \right)$. Then $k_\theta$ and $v_\theta$ can be estimated adaptively over time. At iteration $t$, suppose the adaptive filters produce $k_t$ and $\tilde{v}_t$. The gradient can be constructed by $\frac{\partial J_{P_{0}}(g_\theta)}{\partial \theta} \approx \frac{\partial k_t}{\partial \theta} / \sqrt{\tilde{v}_t} - \frac{1}{2} \left( \frac{k_t | \partial \tilde{v}_t}{\partial \theta} / \sqrt{\tilde{v}_t} \right)$. This trick has been also applied to the divergence cost (9). Now we can present the full algorithm of SGM for density estimation in Algorithm 1.

**Algorithm 1: SGM for Density Estimation**

1. Initialize $k_0 \leftarrow 0; v_0 \leftarrow 0; t \leftarrow 0; \text{Initialize} \ \theta_0$
2. **while** $\theta$ not converge **do**
3. \hspace{1em} $t \leftarrow t + 1$
4. \hspace{1em} Sample $\{c_1, c_2, \ldots, c_M\}$ from $U(0, 1)$; Sample $\{x_1, x_2, \ldots, x_M\}$ from data distribution
5. \hspace{1em} **for** $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, M$ **do**
6. \hspace{2em} $w_{i,j} = f^{(u)}_{\theta_{i-1}}(z_i, c_j); m_{i,j} = f^{(m)}_{\theta_{i-1}}(z_i, c_j); A_{i,j} = f^{(A)}_{\theta_{i-1}}(z_i, c_j)$
7. \hspace{1em} **end for**
8. \hspace{1em} $v_t = \frac{1}{M^2 N^2} \sum_{i,j=1}^N w_{i,j} w_{i,j} \mathcal{N}(m_{i,j} - m_{i',j'}, A_{i,j})$
9. \hspace{1em} $k_t' = \frac{1}{M^2 N^2} \sum_{i,j=1}^N w_{i,j} \mathcal{N}(m_{i,j} - x_j, A_{i,j})$
10. \hspace{1em} $v_t = \beta_t \cdot v_{t-1} + (1 - \beta_t) \cdot v_t'$
11. \hspace{1em} $v_t' = v_t / (1 - \beta_t)$
12. \hspace{1em} $k_t = \beta_t \cdot k_{t-1} + (1 - \beta_t) k_t'$
13. \hspace{1em} $k_t' = k_t / (1 - \beta_t)$
14. \hspace{1em} $\theta_t = \theta_{t-1} + r \left( \frac{\partial k_t}{\partial \theta} / \sqrt{v_t'} - \frac{1}{2} \left( \frac{k_t | \partial \tilde{v}_t}{\partial \theta} / \sqrt{\tilde{v}_t} \right) \right)$
15. **end while**
16. Compute $\tilde{H}_2(p) = -2 \log(k_t / \sqrt{v_t})$

**Modeling conditional probabilities:** Furthermore, SGM is capable of modeling conditional probabilities. We first derive the corresponding variational cost function. Let a conditional probability $p(y|x)$ and an arbitrary $p(x)$ be given. We define
\[ J_{Y|X}(T) = \frac{\int_{X \times Y} T(y|x)p(y|x)p(x) d\mu}{(\int_{X \times Y} T^2(y|x)p(x) d\mu)^{\frac{1}{2}}}. \tag{15} \]

Observe that its upper bound is
\[ J_{Y|X}(T) = \left( \frac{\mathbb{E}_x [T(y|x)p(x)]}{\left( \int T(y|x)p(x) d\mu \right)^{\frac{1}{2}}} \right)^2 \]
\[ \leq \left( \int T(y|x)p(x) d\mu \right)^2. \tag{16} \]

The RHS of (16) is related to Rényi’s definition of conditional entropy (24) as $H_2(Y|X) = -2 \log \text{sup}_{T \in T} J_{Y|X}(T)$. The corresponding solution is $T_0(x|y) = p(y|x)$. To use $J_{Y|X}(T)$ as a cost function for SGM, we assume that $T_0(x|y)$ is approximated by a neural network $g_\theta(y|x)$ by maximizing $J_{Y|X}(T)$. With any given $x$, the network output $g_\theta(y|x)$ defines an IMoG for the variable $y$. Let the network output be $(w_1(c), m_1(c), A_1(c); x)$, it follows that
\[ N_{i,j} = \mathcal{N}(m_i(c_1) - m_j(c_2), A_i(c_1) + A_j(c_2)) \]
\[ \int g_\theta^2(y|x)p(x) d\mu = E_{c_1,c_2} \left[ \frac{1}{N^2} \sum_{i=1}^N w_i(c_1) w_j(c_2) N_{i,j} \right], \tag{17} \]
which has very little difference from the density estimation implementation. Then we maximize $J_{Y|X}(g_\theta)$ as the cost function. The details of the algorithm are shown as Algorithm 2.

**Algorithm 2: SGM for Conditional Distribution Modeling**

1. Initialize $k_0 \leftarrow 0; v_0 \leftarrow 0; t \leftarrow 0; \text{Initialize} \ \theta_0$
2. **while** $\theta$ not converge **do**
3. \hspace{1em} $t \leftarrow t + 1$
4. \hspace{1em} Sample $\{c_1, \ldots, c_M\}$ and $\{c'_1, \ldots, c'_M\}$ from $U(0, 1)$
5. \hspace{1em} Sample $\{z_1, \ldots, z_M\}$ and $\{z'_1, \ldots, z'_M\}$ from Cat($N$)
6. \hspace{1em} Sample $\{x_1, y_1, \ldots, x_M, y_M\}$ from data distribution
7. **for** $i = 1, 2, \ldots, M$ **do**
8. \hspace{2em} $w_i = f^{(u)}_{\theta_{i-1}}(z_i, c_i); m_i = f^{(m)}_{\theta_{i-1}}(z_i, c_i); A_i = f^{(A)}_{\theta_{i-1}}(z_i, c_i)$
9. \hspace{2em} $w'_i = f^{(u)}_{\theta_{i-1}}(z'_i, c'_i); m'_i = f^{(m)}_{\theta_{i-1}}(z'_i, c'_i); A'_i = f^{(A)}_{\theta_{i-1}}(z'_i, c'_i)$
10. **end for**
11. \hspace{1em} $v_t = \frac{1}{M^2} \sum_{i=1}^M w_i w_i' \mathcal{N}(m_i - m_i', A_i + A_i')$
12. \hspace{1em} $k_t' = \frac{1}{M^2} \sum_{i=1}^M w_i w_i' \mathcal{N}(m_i - y_i, A_i + A_i')$
13. \hspace{1em} $\tilde{v}_t = ADP(v_t); \tilde{k}_t = ADP(k_t')$
14. \hspace{1em} $\theta_t = \theta_{t-1} + r(\frac{\partial k_t'}{\partial \theta} / \sqrt{\tilde{v}_t} - \frac{1}{2} (\frac{\tilde{k}_t \partial \tilde{v}_t}{\partial \theta} / \sqrt{\tilde{v}_t})))$
15. **end while**
16. Compute $\tilde{H}_2(Y|X) = -2 \log(\tilde{k}_t / \sqrt{\tilde{v}_t})$

**Experiments**

We show our experiments in two folds. First, we show how the proposed cost functions and SGM provides new opportunities to train a neural network, such as training a mutual
information estimator and a GAN. Secondly, we show the powerful SGM for density estimation and how it differs and improves the conventional GMM and VBGMM.

**Training Neural Networks with Divergence**

In this section, we first show the results of using the divergence cost function [9] to train neural networks, including training a mutual information estimator and a GAN.

**SGM as a mutual information (MI) estimator:** We demonstrate our methods by comparing the performance of estimating MI: (a) SGM density estimator producing the quadratic mutual information (QMI) \( I_Q = -\log_2 (p(x, y)p(y)/p(x)p(y)) + \frac{1}{2} \log_2 (p(x)p(y), p(x,y)) + \frac{1}{2} \log_2 (p(x,y), p(x,y)) \)

(b) Mutual information neural estimation (MINE) [26] producing Rényi’s mutual information; (c) Training the same mapper as MINE with our proposed variational cost \( J_{p,q}(T_\theta) \), which yields MI with Rényi’s second-order divergence (Rényi’s MI).

To estimate \( I_Q \) with SGM, we first construct the probability ratio of the model \( r_\theta(x, y) = \frac{g_\theta(x,y)}{g_\theta(x)g_\theta(y)} \).

We define a function of the probability ratio that evaluates MI of the model given data as \( I_Q(r_\theta) = -\frac{1}{2} \log_2 \left( \frac{\mathbb{E}_{r_\theta} \mathbb{E}_Y [r_\theta(x, y)]}{\mathbb{E}_X \mathbb{E}_Y [r_\theta(x, y)]} \right) \).

It can be verified \( \hat{I}(r_\theta) \) is an unbiased estimator of \( I_Q \) as \( r_\theta \to \frac{p(x,y)}{p(x)p(y)} \). Each term in \( \hat{I}(r_\theta) \) is a byproduct from optimization.

We generate a mixture distribution with 20 centers uniformly sampled from 0.2 to 0.8 in 2D space. We assign each center to a Gaussian distribution with diagonal covariance matrices from \( U(0.0001, 0.002) \). We sample the weighting factor from \( U(0.5, 1.5) \). We set \( N = 300 \) across the experiments.

We test three approaches for the MoG example as before and estimate the mutual information between two dimensions. Figure 1a shows both our approaches are far more stable than MINE which will diverge for small number of samples. Even with variance reduction, MINE suffers from a large variance that cannot be ignored. We compute the ground truth of Rényi’s MI also converges at the same rate, but does not converge to the same value because it uses Rényi’s divergence, instead of QMI, which is a Cauchy-Schwarz divergence [27] estimated with Rényi’s second-order entropy. This also shows that the proposed cost can be applied to any neural network. MINE with enough samples also converges to Shannon’s mutual information but with a high variance. So these proposed estimators are much more well behaved and data efficient to estimate MI, outperforming MINE.

To show the scalability to high dimensions, we generate MoG in high dimensions and compare MINE with optimizing \( J_{p,q}(T_\theta) \), as shown in Figure 1b. This shows our new cost has the same performance with less variance, but much more stable when only a limited number of samples are accessible, as demonstrated in Figure 1a.
class probability [19], entropy, Mahalanobis distance [28], likelihood ratio [27]. Additionally, we also compare with two state-of-the-art methods based on variational information bottleneck (VIB) [29] and nonlinear information bottleneck (NIB) [30], respectively. In terms of metrics, we use the area under ROC curve (AUROC), the area under precision-recall curve (AUPRC), and the false positive rate at 80% true positive (FPR80 ↓). The quantitative results are summarized in Table 1.

Figure 2: Network outputs trained by (18). (a) shows the generated samples. (b) shows the FP and FN samples when the ratio is set to $10^{-2}$. (c) and (c) show the produced probability ratio by the discriminator network for generated data, Fashion MNIST test set, and MNIST test set.

Table 1: AUROC↑, AUPRC↑, and FPR (80%TPR) ↓ for detecting OOD inputs with density ratio and other baseline↓ on Fashion-MNIST vs. MNIST datasets. ↑ indicates that larger value is better, ↓ indicates that lower value is better.

| Method                     | AUROC↑ | AUPRC↑ | FPR80 ↓ |
|----------------------------|--------|--------|----------|
| Density ratio (ours)       | 0.997  | 0.997  | 0.004    |
| p(y|x)                     | 0.734  | 0.702  | 0.506    |
| Entropy of p(y|x)          | 0.746  | 0.726  | 0.448    |
| Mahalanobis distance       | 0.942  | 0.928  | 0.088    |
| Likelihood ratio           | 0.994  | 0.993  | 0.001    |
| VIB                        | 0.906  | 0.903  | 0.172    |
| NIB                        | 0.916  | 0.913  | 0.152    |

Evaluation: We use $J_p(g_0)$ (CE) and the validation rate for evaluation. We compute CE with [5] for GMM, with [12] for SGM and VBGMM. A greater CE indicates a better performance. To address the overfitting issue of GMM, suppose $m'(j)$ is the $j$-th true center, we use the validation rate $VR = E[w \cdot \mathbb{I}(\|m(x) - m'(j)\|_2 < d)/E[w]$ to calculate the percentage of the components whose mean’s distance to the closet true center mass is under $d = 10^{-4}$ or $10^{-2}$.

Performance: Table 2 shows that the performance of SGM is higher than both GMM and VBGMM. If we estimate CE’s upper bound [14] with the true pdf by subintervals, we obtain 6.68, 21.0, 12.5, 7.26 and 8.25. Since only finite samples are given, the true values are higher. Although VBGMM has a better concentration, it introduces a huge bias regarding performance. With $d$ chosen to be $10^{-2}$, all the means of SGM are valid while GMM suffers from severe overfitting, illustrated as Figure 4.

Figure 3: (a) to (d) show the density and model mean components learned by GMM. (e) to (h) show the same figures produced by SGM. The red dots represent the model means and the blue dots represent the data. SGM has a better performance regarding tail.

Table 2: Performance of the 2D density estimation. Although VBGMM has a better concentration, it suffers from a much higher bias compared to vanilla GMM and SGM.

Density Estimation

Now we show how density estimation performance of SGM. We extend the MoG example for mutual information estimations to a generalized mixtures. We randomly assign each center with a Gaussian distribution, a Laplacian distribution, or a uniform distribution. We sample the scale of the Laplacian from $U(0.01, 0.2)$, the subinterval length of each uniform distribution from $U(0.01, 0.2)$, and the weighting factor from $U(0.5, 1.5)$. We generate 200k samples in total and repeat the experiment with different parameters for five times. We compare SGM with GMM and VBGMM. VBGMM has the concentration factor 1.
Modeling Conditional Probabilities

Next we show that SGM is capable of modeling conditional probabilities. We assume that the conditional is given by an artificial continuous-state-space Markov chain. We create a $10 \times 10$ matrix with main diagonals to be 0.7 and the other entries to be 1/30. Then we shuffle the matrix along the $x$ axis and put Gaussian distributions on top of each center, with diagonal covariance matrices sampled from $U(0.0005, 0.002)$ to formulate the single transition probability as $p(x_{t+1}|x_t) = p(x_t, x_{t+1})/\int_X p(x_t)d\mu$. We start with sampling 100 points from a uniform distribution between 0 and 1. Then we generate trajectories with a length of 1000. In other words, samples will have a joint distribution $p(x, y) = p(x)p(x_{t+1} = y|x_t = x)$ with the marginal $p(x) = \mathbb{E}_{x_0} [\frac{1}{T} \sum_{t=0}^T p(x_t = x)]$.

Evaluations: We compare SGM with GMM, VBGM, MDN and conditional GAN (CGAN). For GMM and VBGM, we estimate the joint $p(x, y)$ as described and then marginalize $x$ to obtain $p(y|x) = p(x, y)/\int_{x} p(x, y)d\mu$. For MDN we use the implementation in [5]. We also implement CGAN [10]. Given two pairs $\{x_t, x_{t+1}\}$ and $\{x'_t, x'_{t+1}\}$, the discriminator tries to distinguish between $\{x_t, g(y|x_t)\}$ and $\{x'_t, x'_{t+1}\}$. SGM is trained by [15]. For GMM and VBGM, suppose the covariance matrices are diagonal, $\sum_{i=1}^N w_i N(x - \mu_{x_i}, \sigma_{x_i})N(y - \mu_{y}, \sigma_{y})$. We notice that $g(y|x) = \sum_{i=1}^N w_i N(x - \mu_{x_i}, \sigma_{x_i})N(y - \mu_{y}, \sigma_{y})$. So we first compute the responsibility for each $x \sim p$, then we compute $J_{Y|X}(g_y)$ with [15].

Performance: Figure 4 shows the performance of each method. We find that similar to GMM, MDN also has the tendency to overestimate tail. Since the original joint distribution is a MoG, VBGM is the most competitive but still has visible bias. For CGAN, we use the empirical estimator to produce the joint distribution as Figure 5 (a) and the conditional as Figure 5 (b). Since CGAN does not assume the shape of the distribution and its input noise is uniformly distributed, the estimation will always have a bias. Table 3 shows the overall score evaluated with CE.

Conclusion

This paper presents a novel framework for generative modeling, named SGM, exploiting the nice properties of Rényi’s information in estimation. Since mixtures of Gaussians can be estimated in closed-form by Rényi’s quadratic information, we derive two new cost functions that substitute the cross-entropy and the KL divergence by the corresponding Rényi’s formulations. We then proceed to train a neural network that directly yields a generative model of the input data by local approximations of mean and variance in the data space, e.g., under an infinite Gaussian mixture model assumption. This avoids the issue of using bounds (upper or lower) in the conventional generative modeling approaches, which requires assumptions that are very difficult to fulfill in parametric density modeling. The proposed SGM is data-efficient unlike the conventional approaches as we clearly show with the comparison with MINE. In the experiments, we verified the accuracy of the proposed SGM approach. Further work will be directed towards finding alternate approximations to $T(x)$, which will scale better for high dimensions, and extend the method to alpha close to 1 to estimate Shannon information.

![Figure 4: Comparisons of learned transition probabilities. The red dots show the transition with the highest probability at each state when constructing the Markov chain. (g) shows the joint distribution learned by CGAN. (h) shows $p(x_{t+1}|p_t)$ by taking the network input to be evenly spaced between 0 and 1.](image)

Table 3: Performance for conditional probability modeling.

| Algorithm | EXP #1 | EXP #2 | EXP #3 | EXP #4 | EXP #5 |
|-----------|--------|--------|--------|--------|--------|
| CGAN      | 0.788  | 0.796  | 0.675  | 0.659  | 0.887  |
| GMM       | 0.900  | 0.939  | 0.878  | 0.957  | 0.980  |
| VBGM      | 0.959  | 0.942  | 0.904  | 0.945  | 0.951  |
| MDN       | 0.967  | 0.957  | 0.889  | 0.913  | 0.912  |
| SGM       | 0.993  | 0.968  | 0.923  | 0.972  | 0.990  |

References

[1] José C. Príncipe. Information theoretic learning: Rényi’s entropy and kernel perspectives. Springer Science & Business Media, 2010.
[2] Luis Gonzalez Sanchez Giraldo, Murali Rao, and José C Príncipe. Measures of entropy from data using infinitely divisible kernels. IEEE Transactions on Information Theory, 61(1):535–548, 2014.
[3] Shujian Yu, Luis Gonzalez Giraldo, and José C Príncipe. Information-theoretic methods in deep neural networks: Recent advances and emerging opportunities. In International Joint Conference on Artificial Intelligence, Survey Track, 2021.
[4] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In Proceedings of the 17th International Conference on Neural Information Processing Systems, pages 529–536, 2004.
[5] Christopher M Bishop. Mixture density networks. Technical report, 1994.
[6] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. On surrogate loss functions and f-divergences. The Annals of Statistics, 37(2):876–904, 2009.
[7] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likeli-
hood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010.

[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[9] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[10] Carl Edward Rasmussen et al. The infinite gaussian mixture model. In *Advances in Neural Information Processing Systems*, volume 12, pages 554–560, 1999.

[11] Yingzhen Li and Richard E Turner. Rényi divergence variational inference. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 1081–1089, 2016.

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

[13] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: training generative neural samplers using variational divergence minimization. In *International Conference on Neural Information Processing Systems*, pages 271–279, 2016.

[14] Liam Paninski. Estimation of entropy and mutual information. *Neural Computation*, 15(6):1191–1253, 2003.

[15] John Paisley, David Blei, and Michael I. Jordan. Variational bayesian inference with stochastic search. *International Conference on Machine Learning*, 2012.

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2013.

[17] Alfréd Rényi. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California, 1961.

[18] Kittipat Kampa, Erion Hasanbelliu, and José C Príncipe. Closed-form cauchy–schwarz pdf divergence for mixture of gaussians. In *International Joint Conference on Neural Networks*, pages 2578–2585, 2011.

[19] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

[20] Chi Jin, Yuchen Zhang, Sivaraman Balakrishnan, Martin J Wainwright, and Michael I Jordan. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. *Advances in Neural Information Processing Systems*, pages 4123–4131, 2016.

[21] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Fundamentals of convex analysis*. Springer Science & Business Media, 2004.

[22] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International Conference on Machine Learning*, pages 531–540, 2018.

[23] Bo Hu and José C. Príncipe. Training a bank of wiener models with a novel quadratic mutual information cost function. In *International Conference on Acoustics, Speech and Signal Processing*, pages 3150–3154, 2021.

[24] Alfred Rényi. Some fundamental questions of information theory. *Selected Papers of Alfred Rényi*, 2(174):526–552, 1976.

[25] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International Conference on Machine Learning*, pages 531–540, 2018.

[26] Robert Jenssen, Jose C Principe, Deniz Erdogmus, and Torbjørn Eltoft. The cauchy–schwarz divergence and parzen windowing: Connections to graph theory and mercer kernels. *Journal of the Franklin Institute*, 343(6):614–629, 2006.

[27] Jie Ren, Peter J. Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark Depristo, Joshua Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. 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.

[28] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Neural Information Processing Systems*, 2018.

[29] Alexander A Alemi, Ian Fischer, and Joshua V Dillon. Uncertainty in the variational information bottleneck. In *UAI Uncertainty in Deep Learning Workshop*, 2018.

[30] Artemy Kolchinsky, Brendan D Tracey, and David H Wolpert. Nonlinear information bottleneck. *Entropy*, 21(12):1181, 2019.

[31] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International Conference on Machine Learning*, pages 531–540, 2018.