AR-DAE: Towards Unbiased Neural Entropy Gradient Estimation

Jae Hyun Lim [1 2] **Aaron Courville** [1 2 3 4] **Christopher Pal** [1 5 4] **Chin-Wei Huang** [1 2]

Abstract

Entropy is ubiquitous in machine learning, but it is in general intractable to compute the entropy of the distribution of an arbitrary continuous random variable. In this paper, we propose the *amortized residual denoising autoencoder* (AR-DAE) to approximate the gradient of the log density function, which can be used to estimate the gradient of entropy. Amortization allows us to significantly reduce the error of the gradient approximator by approaching asymptotic optimality of a regular DAE, in which case the estimation is in theory unbiased. We conduct theoretical and experimental analyses on the approximation error of the proposed method, as well as extensive studies on heuristics to ensure its robustness. Finally, using the proposed gradient approximator to estimate the gradient of entropy, we demonstrate state-of-the-art performance on density estimation with variational autoencoders and continuous control with soft actor-critic.

1. Introduction

Entropy is an information theoretic measurement of uncertainty that has found many applications in machine learning. For example, it can be used to incentivize exploration in reinforcement learning (RL) (Haarnoja et al., 2017; 2018); prevent mode-collapse of generative adversarial networks (GANs) (Balaji et al., 2019; Dieng et al., 2019); and calibrate the uncertainty of the variational distribution in approximate Bayesian inference. However, it is in general intractable to compute the entropy of an arbitrary random variable.

In most applications, one actually does not care about the quantity of entropy itself, but rather how to manipulate and control this quantity as part of the optimization objective. In light of this, we propose to approximately estimate the gradient of entropy so as to maximize or minimize the entropy of a data sampler. More concretely, we approximate the gradient of the log probability density function of the data sampler. This is sufficient since the gradient of its entropy can be shown to be the expected value of the *path derivative* (Roeder et al., 2017). We can then plug in a gradient approximator to enable stochastic backpropagation.

We propose to use the *denoising autoencoder* (DAE, Vincent et al. (2008)) to approximate the gradient of the log density function, which is also known as *denoising score matching* (Vincent, 2011). It has been shown that the optimal reconstruction function of the DAE converges to the gradient of the log density as the noise level σ approaches zero (Alain & Bengio, 2014). In fact, such an approach has been successfully applied to recover the gradient field of the density function of high-dimensional data such as natural images (Song & Ermon, 2019), which convincingly shows DAEs can accurately approximate the gradient. However, in the case of entropy maximization (or minimization), the non-stationarity of the sampler's distribution poses a problem for optimization. On the one hand, the log density gradient is recovered only asymptotically as $\sigma \to 0$. On the other hand, the training signal vanishes while a smaller noise perturbation is applied, which makes it hard to reduce the approximation error due to suboptimal optimization. The fact that the sampler's distribution is changing makes it even harder to select a noise level that is sufficiently small. Our work aims at resolving this no-win situation.

In this work, we propose the *amortized residual denoising autoencoder* (AR-DAE), which is a conditional DAE of a residual form that takes in σ as input. We condition the DAE on $\sigma = 0$ at inference time to approximate the log density gradient while sampling non-zero σ at training, which allows us to train with σ sampled from a distribution that covers a wide range of values. If AR-DAE is optimal, we expect to continuously generalize to $\sigma = 0$ to recover the log density gradient, which can be used as an unbiased estimate of the entropy gradient. We perform ablation studies on the approximation error using a DAE, and show that our method provides significantly more accurate approximation than the baselines. Finally, we apply our method to improve distribution-free inference for variational autoen-

[1]Mila [2]Université de Montréal [3]CIFAR fellow [4]Canada CIFAR AI Chair [5]Polytechnique Montréal. Correspondence to: Jae Hyun Lim <jae.hyun.lim@umontreal.ca>, Chin-Wei Huang <chinwei.huang@umontreal.ca>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

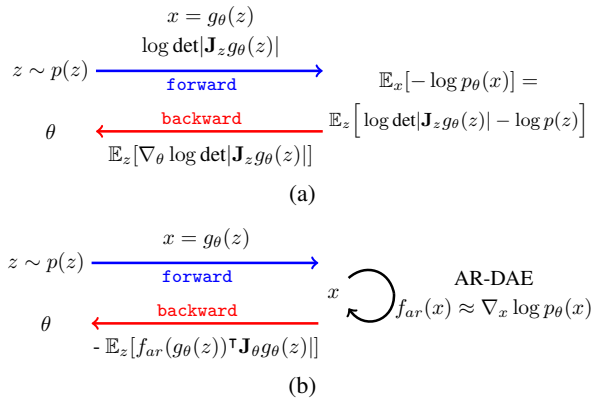

Figure 1. (a) Entropy gradient wrt parameters of an invertible generator function. (b) Approximate entropy gradient using the proposed method.

coders (Kingma & Welling, 2014; Rezende et al., 2014b) and soft actor-critic (Haarnoja et al., 2018) for continuous control problems in reinforcement learning. As these tasks are non-stationary, amortized (conditional), and highly structured, it demonstrates AR-DAE can robustly and accurately approximate log density gradient of non-trivial distributions given limited computational budgets.

2. Approximate entropy gradient estimation

2.1. Background on tractability of entropy

An *implicit density model* is characterized by a data generation process (Mohamed & Lakshminarayanan, 2016). The simplest form of an implicit density model contains a prior random variable $z \sim p(z)$, and a generator function $g : z \mapsto x$. The likelihood of a particular realization of x is *implied* by the pushforward of $p(z)$ through the mapping g.

Unlike an *explicit density model*, an implicit density model does not require a carefully designed parameterization for the density to be explicitly defined, allowing it to approximate arbitrary data generation process more easily. This comes at a price, though, since the density function of the implicit model cannot be easily computed, which makes it hard to approximate its entropy using Monte Carlo methods.

2.2. Denoising entropy gradient estimator

Let z and g be defined as above, and let θ be the parameters of the mapping g (denoted g_θ). Most of the time, we are interested in maximizing (or minimizing) the entropy of the implicit distribution of $x = g_\theta(z)$. For example, when the mapping g is a bijection, the density of $x = g_\theta(z)$ can be decomposed using the change-of-variable density formula, so controlling the entropy of x amounts to controlling the log-determinant of the Jacobian of g_θ (Rezende & Mohamed, 2015), as illustrated in Figure 1-(a). This allows us to estimate both the entropy and its gradient. However, for an iterative optimization algorithm such as (stochastic) gradient descent, which is commonly employed in machine learning, it is sufficient to compute the gradient of the entropy rather than the entropy itself.

Following Roeder et al. (2017), we can rewrite the entropy of x by changing the variable and neglecting the score function which is 0 in expectation to get

$$\nabla_\theta H(p_g(x)) = -\mathbb{E}_z\left[[\nabla_x \log p_g(x)|_{x=g_\theta(z)}]^\intercal \mathbf{J}_\theta g_\theta(z)\right], \tag{1}$$

where $\mathbf{J}_\theta g_\theta(z)$ is the Jacobian matrix of the random sample $x = g_\theta(z)$ wrt to the sampler's parameters θ. See Appendix A for the detailed derivation. We emphasize that this formulation is more general as it does not require g to be bijective.

Equation (1) tells us that we can obtain an unbiased estimate of the entropy by drawing a sample of the integrand, which is the *path derivative* of z. The integrand requires evaluating the sample $x = g_\theta(z)$ under the gradient of its log density $\nabla_x \log p_g(x)$. As $\log p_g(x)$ is usually intractable or simply not available, we directly approximate its gradient using a black box function. As long as we can provide a good enough approximation to the gradient of the log density and treat it as the incoming unit in the backward differentiation (see Figure 1-(b)), the resulting estimation of the entropy gradient is approximately unbiased.

In this work, we propose to approximate the gradient of the log density using a denoising autoencoder (DAE, Vincent et al. (2008)). A DAE is trained by minimizing the reconstruction loss d of an autoencoder r with a randomly perturbed input

$$\mathcal{L}_{\texttt{DAE}}(r) = \mathbb{E}[d(x, r(x + \epsilon))],$$

where the expectation is taken over the random perturbation ϵ and data x. Alain & Bengio (2014) showed that if d is the L2 loss and ϵ is a centered isotropic Gaussian random variable with variance σ^2, then under some mild regularity condition on $\log p_g$ the optimal reconstruction function satisfies

$$r^*(x) = x + \sigma^2 \nabla_x \log p_g(x) + o(\sigma^2),$$

as $\sigma^2 \to 0$. That is, for sufficiently small σ, we can approximate the gradient of the log density using the black box function $f_r(x) := \frac{r(x)-x}{\sigma^2}$ assuming $r \approx r^*$.

3. Error analysis of $\nabla_x \log p_g(x) \approx f_r(x)$

Naively using $f_r(x)$ to estimate the gradient of the entropy is problematic. First of all, the division form of f_r can lead to numerical instability and magnify the error of approximation. This is because when the noise perturbation σ is small, $r(x)$ will be very close to x and thus both the numerator and the denominator of f_r are close to zero.

Second, using the triangle inequality, we can decompose the error of the approximation $\nabla_x \log p_g(x) \approx f_r(x)$ into

$$||\nabla_x \log p_g(x) - f_r(x)|| \leq \underbrace{||\nabla_x \log p_g(x) - f_{r^*}(x)||}_{\textit{asymp error}} + ||f_{r^*}(x) - f_r(x)||.$$

The first error is incurred by using the optimal DAE to approximate $\nabla_x \log p_g(x)$, which vanishes when $\sigma \to 0$. We refer to it as the *asymptotic error*. The second term is the difference between the optimal DAE and the "current" reconstruction function. Since we use a parametric family of functions (denoted by $\mathcal{F}$) to approximate f_{r^*}, it can be further bounded by

$$||f_{r^*}(x) - f_r(x)|| \leq \underbrace{||f_{r^*}(x) - f_{r^*_{\mathcal{F}}}(x)||}_{\textit{param error}} + \underbrace{||f_{r^*_{\mathcal{F}}}(x) - f_r(x)||}_{\textit{optim error}},$$

where $r^*_{\mathcal{F}} := \arg\min_{r \in \mathcal{F}} \mathcal{L}_{\texttt{DAE}}(r)$ is the optimal reconstruction function within the family $\mathcal{F}$. The first term measures how closely the family of functions $\mathcal{F}$ approximates the optimal DAE, and is referred to as the *parameterization error*. The second term reflects the suboptimality in optimizing r. It can be significant especially when the distribution of x is non-stationary, in which case r needs to be constantly adapted. We refer to this last error term as the *optimization error*. As we use a neural network to parameterize r, the parameterization error can be reduced by increasing the capacity of the network. The optimization error is subject to the variance of the noise σ^2 (relative to the distribution of x), as it affects the magnitude of the gradient signal $\mathbb{E}[\nabla||r(x+\epsilon) - x||^2]$. This will make it hard to design a fixed training procedure for r as different values of σ requires different optimization specifications to tackle the optimization error.

4. Achieving asymptotic optimality

In this section, we propose the **amortized residual DAE** (AR-DAE), an improved method to approximate $\nabla_x \log p_g(x)$ that is designed to resolve the numerical instability issue and reduce the error of approximation.

4.1. Amortized residual DAE

AR-DAE (denoted f_{ar}) is a DAE of residual form conditioned on the magnitude of the injected noise, minimizing the following optimization objective.

$$\mathcal{L}_{\texttt{ar}}(f_{ar}) = \mathop{\mathbb{E}}_{\substack{x\sim p(x) \\ u\sim N(0,I) \\ \sigma\sim N(0,\delta^2)}} \left[\left\| u + \sigma f_{ar}(x + \sigma u; \sigma) \right\|^2 \right]. \quad (2)$$

This objective involves three modifications to the regular training and parameterization of a DAE: *residual connection*, *loss rescaling*, and *scale conditioning* for amortization.

Residual form First, we consider a residual form of DAE (up to a scaling factor): let $r(x) = \sigma^2 f_{ar}(x) + x$, then $\nabla_x \log p_g(x)$ is approximately equal to

$$\frac{r(x) - x}{\sigma^2} = \frac{\sigma^2 f_{ar}(x) + x - x}{\sigma^2} = f_{ar}\,.$$

That is, this reparameterization allows f_{ar} to directly approximate the gradient, avoiding the division that can cause numerical instability. The residual form also has an obvious benefit of a higher capacity, as it allows the network to represent an identity mapping more easily, which is especially important when the reconstruction function is close to an identity map for small values of σ (He et al., 2016).

Loss rescaling To prevent the gradient signal from vanishing to 0 too fast when σ is arbitrarily small, we rescale the loss $\mathcal{L}_{\texttt{DAE}}$ by a factor of $1/\sigma$, and since we can decouple the noise level from the isotropic Gaussian noise into $\epsilon = \sigma u$ for standard Gaussian u, the rescaled loss can be written as $\mathbb{E}[||\sigma f_{ar}(x + \sigma u) + u||^2]$.

We summarize the properties of the optimal DAE of the rescaled residual form in the following propositions:

Proposition 1. *Let x and u be distributed by $p(x)$ and $\mathcal{N}(0, I)$. For $\sigma \neq 0$, the minimizer of the functional $\mathbb{E}_{x,u}[||u + \sigma f(x + \sigma u)||^2]$ is almost everywhere determined by*

$$f^*(x; \sigma) = \frac{-\,\mathbb{E}_u[p(x - \sigma u)u]}{\sigma\,\mathbb{E}_u[p(x - \sigma u)]}\,.$$

Furthermore, if $p(x)$ and its gradient are both bounded, f^* is continuous wrt σ for all $\sigma \in \mathbb{R}\backslash 0$ and $\lim_{\sigma\to 0} f^*(x; \sigma) = \nabla_x \log p_g(x)$.

The above proposition studies the asymptotic behaviour of the optimal f^*_{ar} as $\sigma \to 0$. Below, we show that under the same condition, f^*_{ar} approaches the gradient of the log density function of a Gaussian distribution centered at the expected value of $x \sim p(x)$ as σ is arbitrarily large.

Proposition 2. $\lim_{\sigma\to\infty} \frac{f^*(x;\sigma)}{\nabla_x \log \mathcal{N}(x; \mathbb{E}_p[X], \sigma^2 I)} \to 1.$

Scale conditioning Intuitively, with larger σ values, the perturbed data $x + \sigma u$ will more likely be "off-manifold", which makes it easy for the reconstruction function to point back to where most of the probability mass of the distribution of x resides. Indeed, as Proposition 2 predicts, with larger σ the optimal f^*_{ar} tends to point to the expected value $\mathbb{E}_p[X]$, which is shown in Figure 2-left. With smaller values of σ, training f_{ar} becomes harder, as one has to predict the vector $-u$ from $x + \sigma u$ (i.e. treating x as noise and trying to recover u). Formally, the training signal (Δ) has a decaying

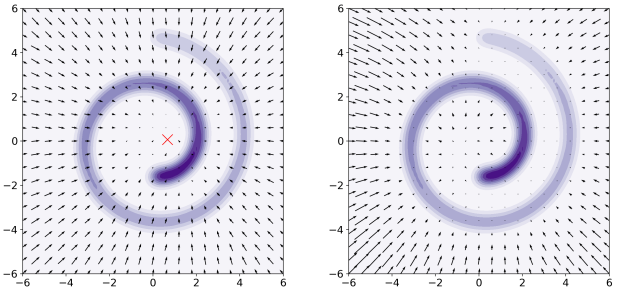

Figure 2. Residual DAE trained with a large (left) vs small (right) σ value. Red cross indicates the mean of the swissroll. The arrows indicate the approximate gradient directions.

rate of $\mathcal{O}(\sigma^2)$ for small σ values, because

$$\mathbb{E}_u[\Delta] := \mathbb{E}_u[\nabla||u + \sigma f(x + \sigma u)||^2] \\ = 2\sigma^2 \nabla \left(\mathbf{tr}(\nabla_x f(x)) + \frac{1}{2}||f(x)||^2 \right) + o(\sigma^2),$$

where the first term is proportional to the stochastic gradient of the *implicit score matching* (Hyvärinen, 2005). That is, with smaller σ values, minimizing the rescaled loss is equivalent to score matching, up to a diminishing scaling factor. Moreover, the variance of the gradient signal $\mathrm{Var}(\Delta)$ also has a quadratic rate $\mathcal{O}(\sigma^2)$, giving rise to a decreasing signal-to-noise ratio (SNR) $\mathbb{E}[\Delta]/\sqrt{\mathrm{Var}(\Delta)} = \mathcal{O}(\sigma)$, which is an obstacle for stochastic optimization (Shalev-Shwartz et al., 2017). See Appendix C for the SNR analysis.

In order to leverage the asymptotic optimality of the gradient approximation as $\sigma \to 0$ (Figure 2-right), we propose to train multiple (essentially infinitely many) models with different σ's at the same time, hoping to leverage the benefit of training a large-σ model while training a model with a smaller σ.

More concretely, we condition f_{ar} on the scaling factor σ, so that f_{ar} can "generalize" to the limiting behaviour of f^*_{ar} as $\sigma \to 0$ to reduce the asymptotic error. Note that we cannot simply take σ to be zero, since setting $\sigma = 0$ would result in either learning an identity function for a regular DAE or learning an arbitrary function for the rescaled residual DAE (as the square loss would be independent of the gradient approximator).

The scale-conditional gradient approximator $f_{ar}(x; \sigma)$ will be used to approximate $\nabla_x \log p_g(x)$ by setting $\sigma = 0$ during inference, while σ is never zero at training. This can be done by considering a distribution of σ, which places zero probability to the event $\{\sigma = 0\}$; e.g. a uniform density between $[0, \delta]$ for some $\delta > 0$. The issue of having a non-negative support for the distribution of σ is that we need to rely on f_{ar} to extrapolate to 0, but neural networks usually perform poorly at extrapolation. This can be resolved by having a symmetric distribution such as centered Gaussian

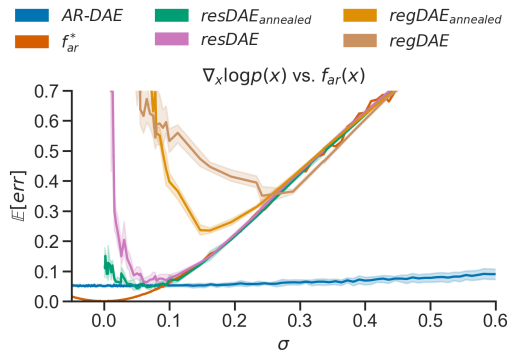

Figure 3. Approximating log density gradient of 1-D MoG. *AR-DAE*: the approximation error of the proposed method. f^*_{ar}: the optimal DAE. *resDAE*: a DAE of residual form (as well as loss rescaling). *regDAE*: a regular DAE.

with variance δ^2 or uniform density between $[-\delta, \delta]$; owing to the the symmetry of the noise distribution $N(u; 0, I)$, we can mirror the scale across zero without changing the loss:

$$\mathbb{E}_u \left[||u + \sigma f(x + \sigma u)||^2 \right] = \mathbb{E} \left[||(-u) + \sigma f(x + \sigma(-u))||^2 \right] \\ = \mathbb{E} \left[||u + (-\sigma) f(x + (-\sigma)u)||^2 \right].$$

Furthermore, Proposition 1 implies a good approximation to $f^*_{ar}(x, \sigma')$ would be close to $f^*_{ar}(x, \sigma)$ if σ' is sufficiently close to σ. We suspect this might help to reduce the optimization error of AR-DAE, since the continuity of both f_{ar} and f^*_{ar} implies that $f_{ar}(x, \sigma)$ only needs to refine $f_{ar}(x, \sigma')$ slightly if the latter already approximates the curvature of $f^*_{ar}(x, \sigma')$ well enough. Then by varying different σ values, the conditional DAE is essentially interpolating between the gradient field of the log density function of interest and that of a Gaussian with the same expected value.

4.2. Approximation error

To study the approximation error with different variants of the proposed method, we consider a 1-dimensional mixture of Gaussians (MoG) with two equally weighted Gaussians centered at 2 and -2, and with a standard deviation of 0.5, as this simple distribution has a non-linear gradient function and an analytical form of the optimal gradient approximator f^*. See Appendix D.1 for the formula and an illustration of approximation with f^* with different σ values.

We let p be the density function of the MoG just described. For a given gradient approximator f, we estimate the expected error $\mathbb{E}_p[|\nabla_x \log p(x) - f|]$ using 1000 i.i.d. samples of $x \sim p$. The results are presented in Figure 3. The curve of the expected error of the optimal f^*_{ar} shows the asymptotic error indeed shrinks to 0 as $\sigma \to 0$, and it serves as a theoretical lower bound on the overall approximation error.

Our ablation includes two steps of increments. First, modifying the regular DAE (*regDAE*) to be of the residual form (with loss rescaling, *resDAE*) largely reduces the param-

eterization error and optimization error combined, as we use the same architecture for the reconstruction function of *regDAE* and for the residual function of *resDAE*. We also experiment with annealing the σ values (as opposed to training each model individually): we take the model trained with a larger σ to initialize the network that will be trained with a slightly smaller σ. Annealing significantly reduces the error and thus validates the continuity of the optimal f^*_{ar}. All four curves have a jump in the error when σ gets sufficiently small, indicating the difficulty of optimization when the training signal diminishes. This leads us to our second increment: amortization of training (i.e. AR-DAE). We see that not only does the error of AR-DAE decrease and transition more smoothly as σ gets closer to 0, but it also significantly outperforms the optimal f^*_{ar} for large σ's. We hypothesize this is due to the choice of the distribution over σ; $\mathcal{N}(0, \delta^2)$ concentrates around 0, which biases the training of f_{ar} to focus more on smaller values of σ.

5. Related Works

Denoising autoencoders were originally introduced to learn useful representations for deep networks by Vincent et al. (2008; 2010). It was later on noticed by Vincent (2011) that the loss function of the residual form of DAE is equal to the expected quadratic error $||f - \nabla_x \log p_\sigma||^2$, where $p_\sigma(x') = \int p(x)\mathcal{N}(x'; x, \sigma^2 I)dx$ is the marginal distribution of the perturbed data, to which the author refers as *denoising score matching*. Minimizing expected quadratic error of this form is in general known as *score matching* (Hyvärinen, 2005), where $\nabla_x \log p$ is referred to as the score [1] of the density p. And it is clear now when we convolve the data distribution with a smaller amount of noise, the residual function f tends to approximate $\nabla_x \log p(x)$ better. This is formalized by Alain & Bengio (2014) as the limiting case of the optimal DAE. Saremi et al. (2018); Saremi & Hyvarinen (2019) propose to use the residual and gradient parameterizations to train a deep energy model with denoising score matching.

As a reformulation of score matching, instead of explicitly minimizing the expected square error of the score, the original work of Hyvärinen (2005) proposes the *Implicit score matching* and minimizes

$$\mathbb{E}_p \left[\frac{1}{2} ||f(x)||^2 + \mathbf{tr}(\nabla_x f(x)) \right]. \tag{3}$$

Song et al. (2019) proposed a stochastic algorithm called the *sliced score matching* to estimate the trace of the Jacobian, which reduces the computational cost from $\mathcal{O}(d_x^2)$ to $\mathcal{O}(d_x)$ (where d_x is the dimensionality of x). It was later noted by the same author that the computational cost of the sliced score matching is still much higher than that of the denoising score matching (Song & Ermon, 2019).

[1] This is not to be confused with the score (or informant) in statistics, which is the gradient of the log likelihood function wrt the parameters.

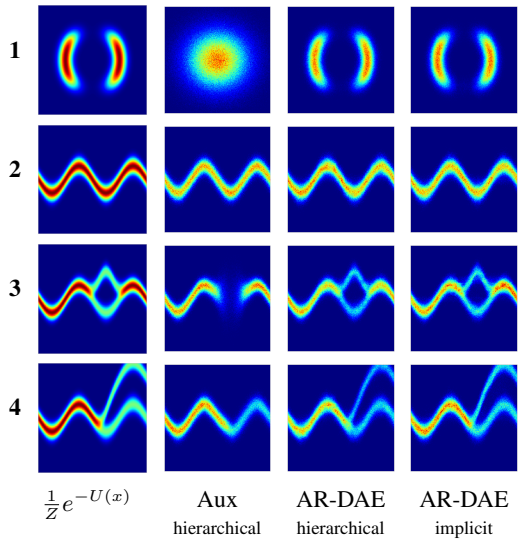

Figure 4. Fitting energy functions. First column: target energy functions. Second column: auxiliary variational method for hierarchical model. Third column: hierarchical model trained with AR-DAE. Last column: implicit model trained with AR-DAE.

Most similar to our work are Song & Ermon (2019) and Bigdeli et al. (2020). Song & Ermon (2019) propose to learn the score function of a data distribution, and propose to sample from the corresponding distribution of the learned score function using Langevin dynamics. They also propose a conditional DAE trained with a sequence of σ's in decreasing order, and anneal the potential energy for the Langevin dynamics accordingly to tackle the mixing problem of the Markov chain. Bigdeli et al. (2020) propose to match the score function of the data distribution and that of an implicit sampler. As the resulting algorithm amounts to minimizing the reverse KL divergence, their proposal can be seen as a combination of Song & Ermon (2019) and our work.

Implicit density models are commonly seen in the context of likelihood-free inference (Mescheder et al., 2017; Tran et al., 2017; Li et al., 2017; Huszár, 2017). Statistics of an implicit distribution are usually intractable, but there has been an increasing interest in approximately estimating the gradient of the statistics, such as the entropy (Li & Turner, 2018; Shi et al., 2018) and the mutual information (Wen et al., 2020).

6. More Analyses and Experiments

6.1. Energy function fitting

In Section 4.2, we have analyzed the error of approximating the gradient of the log-density function in the context of a fixed distribution. In reality, we usually optimize the distribution iteratively, and once the distribution is updated,

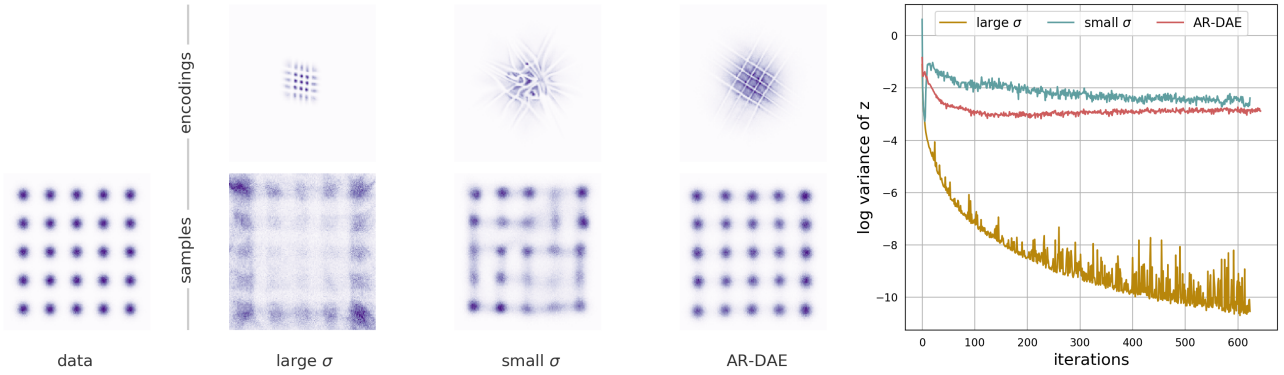

Figure 5. Density estimation with VAE on 5×5 grid MoG. 1st column: data sampled from the MoG. 2nd column: VAE+DAE in residual form trained with a large σ value. 3rd column: VAE+DAE in residual form trained with a small σ value. 4th column: VAE+AR-DAE. Last column: averaged log variance of z throughout training.

the gradient approximator also needs to be updated accordingly to constantly provide accurate gradient signal. In this section, we use the proposed entropy gradient estimator to train an implicit sampler to match the density of some unnormalized energy functions.

Concretely, we would like to approximately sample from a target density which can be written as $p_{\texttt{target}}(x) \propto \exp(-U(x))$, where $U(x)$ is an energy function. We train a neural sampler g by minimizing the reverse Kullback-Leibler (KL) divergence

$$\begin{aligned} &D_{KL}(p_g(x)||p_{\texttt{target}}(x)) \\ &= -H(p_g(x)) + \mathop{\mathbb{E}}_{x\sim p_g(x)}\left[-\log p_{\texttt{target}}(x)\right]. \end{aligned} \tag{4}$$

where p_g is the density induced by g. We use the target energy functions U proposed in Rezende & Mohamed (2015) (see Table 4 for the formulas). The corresponding density functions are illustrated at the first column of Figure 4.

We consider two sampling procedures for g. The first one has a hierarchical structure: let z be distributed by $\mathcal{N}(0, I)$, and x be sampled from the conditional $p_g(x|z) := \mathcal{N}(\mu_g(z), \sigma_g^2(z))$ where μ_g and $\log \sigma_g$ are parameterized by neural networks. The resulting marginal density has the form $p_g(x) = \int p_g(x|z)p_g(z)dz$, which is computationally intractable due to the marginalization over z. We compare against the variational method proposed by Agakov & Barber (2004), which lower-bounds the entropy by

$$H(p_g(x)) \geq -\mathop{\mathbb{E}}_{x,z\sim p_g(x|z)p(z)}\left[\log \frac{p_g(x|z)p(z)}{h(z|x)}\right]. \tag{5}$$

Plugging (5) into (4) gives us an upper bound on the KL divergence. We train p_g and h jointly by minimizing this upper bound [2] as a baseline.

[2]The normalizing constant of the target density will not affect the gradient $\nabla_x \log p_{\texttt{target}}(x) = -\nabla_x U(x)$.

The second sampling procedure has an implicit density: we first sample $z \sim \mathcal{N}(0, I)$ and pass it through the generator $x = g(z)$. We estimate the gradient of the negentropy of both the hierarchical and implicit models by following the approximate gradient of the log density $f_{ar} \approx \log p_g$. The experimental details can be found in Appendix E.

As shown in Figure 4, the density learned by the auxiliary method sometimes fails to fully capture the target density. As in this experiment, we anneal the weighting of the cross-entropy term from 0.01 to 1, which is supposed to bias the sampler to be rich in noise during the early stage of training, the well-known mode seeking behavior of reverse KL-minimization should be largely mitigated. This suggests the imperfection of the density trained with the auxiliary method is a result of the looseness of the variational lower bound on entropy, which leads to an inaccurate estimate of the gradient. On the other hand, the same hierarchical model and the implicit model trained with AR-DAE both exhibit much higher fidelity. This suggests our method can provide accurate gradient signal even when the sampler's distribution p_g is being constantly updated. [3]

6.2. Variational autoencoder

In the previous section, we have demonstrated that AR-DAE can robustly approximate the gradient of the log density function that is constantly changing and getting closer to some target distribution. In this section, we move on to a more challenging application: likelihood-free inference for variational autoencoders (VAE, Kingma & Welling (2014); Rezende et al. (2014a)). Let $p(z)$ be the standard normal. We assume the data is generated by $x \sim p(x|z)$ which is parameterized by a deep neural network. To estimate the parameters, we maximize the marginal likelihood $\int p(x|z)p(z)dz$ of the data x, sampled from some data dis-

[3]We update f_{ar} 5 times per update of p_g to generate this figure; we also include the results with less updates of f_{ar} in Appendix E.

	$\log p(x)$		
	MLP	*Conv*	*ResConv*
Gaussian†	-85.0	-81.9	-
HVI aux†	-83.8	-81.6	-
AVB†	-83.7	-81.7	-
Gaussian	-84.40	-81.82	-80.75
HVI aux	-84.63	-81.87	-80.80
HVI AR-DAE (ours)	**-83.42**	-81.46	-80.45
IVI AR-DAE (ours)	-83.62	**-81.26**	**-79.18**

Table 1. Dynamically binarized MNIST. †Results taken from Mescheder et al. (2017).

Model	$\log p(x)$
(Models with a trained prior)	
VLAE (Chen et al., 2016)	**-79.03**
PixelHVAE + VampPrior (Tomczak & Welling, 2018)	-79.78
(Models without a trained prior)	
VAE IAF (Kingma et al., 2016)	-79.88
VAE NAF (Huang et al., 2018)	-79.86
Diagonal Gaussian	-81.43
IVI AR-DAE (ours)	**-79.61**

Table 2. Statically binarized MNIST.

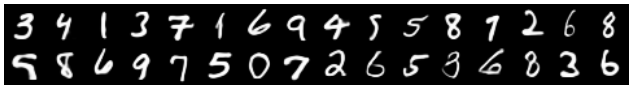

Figure 6. Generated samples (the mean value of the decoder) of the IVI AR-DAE trained on statically binarized MNIST.

tribution $p_{\texttt{data}}(x)$. Since the marginal likelihood is usually intractable, the standard approach is to maximize the *evidence lower bound* (ELBO):

$$\log p(x) \geq \mathop{\mathbb{E}}_{z\sim q(z|x)} \left[\log p(x,z) - \log q(z|x)\right], \quad (6)$$

where $q(z|x)$ is an amortized variational posterior distribution. The ELBO allows us to jointly optimize $p(x|z)$ and $q(z|x)$ with a unified objective.

Note that the equality holds *iff* $q(z|x) = p(z|x)$, which motivates using more flexible families of variational posterior. Note that this is more challenging for two reasons: the target distribution $p(z|x)$ is constantly changing and is conditional. Similar to Mescheder et al. (2017), we parameterize a conditional sampler $z = g(\epsilon, x)$, $\epsilon \sim \mathcal{N}(0, I)$ with an implicit $q(z|x)$. We use AR-DAE to approximate $\nabla_z \log q(z|x)$ and estimate the entropy gradient to update the encoder while maximizing the ELBO. To train AR-DAE, instead of fixing the prior variance δ in the $\mathcal{L}_{ar}$ we adaptively choose δ for different data points. See Appendix F for a detailed description of the algorithm and heuristics we use.

Toy dataset To demonstrate the difficulties in inference, we train a VAE with a 2-D latent space on a mixture of 25 Gaussians. See Appendix F.3 for the experimental details.

In Figure 5, we see that if a fixed σ is chosen to be too large for the residual DAE, the DAE tends to underestimate the gradient of the entropy, so the variational posteriors collapse to point masses. If σ is too small, the DAE manages to maintain a non-degenerate variational posterior, but the inaccurate gradient approximation results in a non-smooth encoder and poor generation quality. On the contrary, the same model trained with AR-DAE has a very smooth encoder that maps the data into a Gaussian-shaped, aggregated posterior and approximates the data distribution accurately.

MNIST We first demonstrate the robustness of our method on different choices of architectures for VAE: (1) a one-hidden-layer fully-connected network (denoted by *MLP*), (2) a convolutional network (denoted by *Conv*), and (3) a larger convolutional network with residual connections (denoted by *ResConv*) from (Huang et al., 2018). The first two architectures are taken from Mescheder et al. (2017) for a direct comparison with the adversarially trained implicit variational posteriors (AVB). We also implement a diagonal Gaussian baseline and the auxiliary hierarchical method (HVI aux, (Maaløe et al., 2016)). We apply AR-DAE to estimate the entropy gradient of the hierarchical posterior and the implicit posterior (denoted by HVI AR-DAE and IVI AR-DAE, respectively). As shown in Table 1, AR-DAE consistently improves the quality of inference in comparison to the auxiliary variational method and AVB, which is reflected by the better likelihood estimates.

We then compare our method with state-of-the-art VAEs evaluated on the statically binarized MNIST dataset (Larochelle & Murray, 2011). We use the implicit distribution with the *ResConv* architecture following the previous ablation. As shown in Table 2, the VAE trained with AR-DAE demonstrates state-of-the-art performance among models with a fixed prior. Generated samples are presented in Figure 6.

6.3. Entropy-regularized reinforcement learning

We now apply AR-DAE to approximate entropy gradient in the context of reinforcement learning (RL). We use the *soft actor-critic* (SAC, Haarnoja et al. (2018)), a state-of-the-art off-policy algorithm for continuous control that is designed to encourage exploration by regularizing the entropy of the policy. We train the policy $\pi(a|s)$ to minimize the following objective:

$$\mathcal{L}(\pi) = \mathop{\mathbb{E}}_{s\sim\mathcal{D}} \left[D_{KL}\left(\pi(a|s) \middle\| \frac{\exp\left(Q(s,a)\right)}{Z(s)}\right)\right],$$

where $\mathcal{D}$ is a replay buffer of the past experience of the agent, Q is a "soft" state-action value function that approximates the entropy-regularized expected return of the policy, and

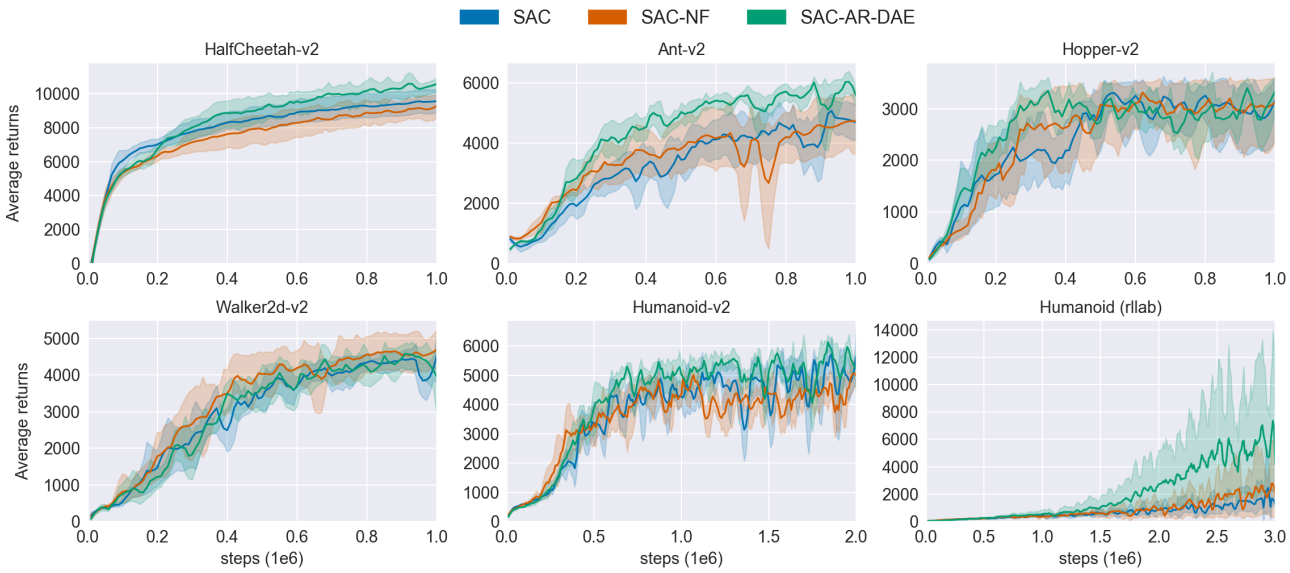

Figure 7. Continuous control in reinforcement learning. SAC: soft actor-critic with diagonal Gaussian. SAC-NF: soft actor-critic with normalizing flows. SAC-AR-DAE: soft actor-critic with implicit distribution trained with AR-DAE. The shaded area indicates the standard error with 5 runs.

$Z(s) = \int_a \exp(Q(s,a))da$ is the normalizing constant of the Gibbs distribution. A complete description of the SAC algorithm can be found in Appendix G.1. We compare with the original SAC that uses a diagonal Gaussian distribution as policy and a normalizing flow-based policy proposed by Mazoure et al. (2019). We parameterize an implicit policy and use AR-DAE to approximate $\nabla_a \log \pi(a|s)$ to estimate

$$\nabla_\phi \mathcal{L}(\pi) = \mathop{\mathbb{E}}_{\substack{s\sim\mathcal{D}\\a\sim\pi}} \left[[\nabla_a \log \pi_\phi(a|s) - \nabla_a Q(s,a)]^\intercal \mathbf{J}_\phi g_\phi(\epsilon, s) \right], \quad (7)$$

where $\pi(a|s)$ is implicitly induced by $a = g_\phi(\epsilon, s)$ with $\epsilon \sim \mathcal{N}(0, I)$. We parameterize f_{ar} as the gradient of a scalar function F_{ar}, so that F_{ar} can be interpreted as the unnormalized log-density of the policy which will be used to update the soft Q-network. We run our experiments on six continuous control environments from the OpenAI gym benchmark suite (Brockman et al., 2016) and Rllab (Duan et al., 2016). The experimental details can be found in Appendix G.2.

The results are presented in Table 3 and Figure 7. We see that SAC-AR-DAE using an implicit policy improves the performance over SAC-NF. This also shows the approximate gradient signal of AR-DAE is stable and accurate even for reinforcement learning. The extended results for a full comparison of the methods are provided in Table 5 and 6.

6.4. Maximum entropy modeling

As a last application, we apply AR-DAE to solve the constrained optimization problem of the *maximum entropy principle*. Let $m \in \mathbb{R}^{10}$ be a random vector and $B \in \mathbb{R}^{10\times 10}$ be a random matrix with m_i and B_{ij} drawn i.i.d. from $\mathcal{N}(0,1)$. It is a standard result that among the class of real-valued random vectors $x \in \mathbb{R}^{10}$ satisfying the constraints $\mathbb{E}[x] = m$ and $\mathrm{Var}(x) = B^\top B$, $x \sim \mathcal{N}(m, B^\top B)$ has the maximal entropy. Similar to Loaiza-Ganem et al. (2017), we solve this constrained optimization problem but with an implicit distribution. We use the *penalty method* and increasingly penalize the model to satisfy the constraints. Concretely, let $\tilde{m}$ and $\tilde{C}$ be the sample mean and sample covariance matrix, respectively, estimated with a batch size of 128. We minimize the modified objective $-H(p_\theta(x)) + \lambda \sum_{j\in\{1,2\}} c_j^2$, where $c_1 = ||\tilde{m} - m||_2$ and $c_2 = ||\tilde{C} - B^\top B||_F$, with

	SAC	SAC-NF	SAC-AR-DAE
HalfCheetah-v2	9695 ± 879	9325 ± 775	$\mathbf{10907 \pm 664}$
Ant-v2	5345 ± 553	4861 ± 1091	$\mathbf{6190 \pm 128}$
Hopper-v2	$\mathbf{3563 \pm 119}$	3521 ± 129	3556 ± 127
Walker-v2	4612 ± 249	4760 ± 624	$\mathbf{4793 \pm 395}$
Humanoid-v2	5965 ± 179	5467 ± 44	$\mathbf{6275 \pm 202}$
Humanoid (rllab)	6099 ± 8071	3442 ± 3736	$\mathbf{10739 \pm 10335}$

Table 3. Maximum average return. $\pm$ corresponds to one standard deviation over five random seeds.

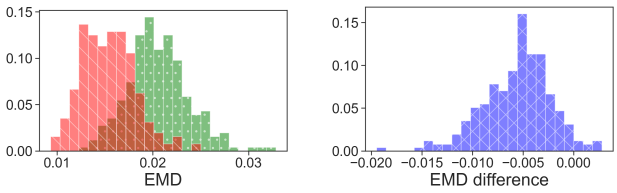

Figure 8. Maximum entropy principle experiment. Left: estimated EMD of (red) the implicit distribution trained with AR-DAE and (green) the IAF. Right: the estimated EMD of the implicit distribution minus that of the IAF.

increasing weighting λ on the penalty. We estimate the entropy gradient using AR-DAE, and compare against the inverse autoregressive flows (IAF, Kingma et al. (2016)). At the end of training, we estimate the earth mover's distance (EMD) from $\mathcal{N}(m, B^\top B)$.

We repeat the experiment 256 times and report the histogram of EMD in Figure 8. We see that most of the time the implicit model trained with AR-DAE has a smaller EMD, indicating the extra flexibility of arbitrary parameterization allows it to satisfy the geometry of the constraints more easily. We leave some more interesting applications suggested in Loaiza-Ganem et al. (2017) for future work.

7. Conclusion

We propose AR-DAE to estimate the entropy gradient of an arbitrarily parameterized data generator. We identify the difficulties in approximating the log density gradient with a DAE, and demonstrate the proposed method significantly reduces the approximation error. In theory, AR-DAE approximates the zero-noise limit of the optimal DAE, which is an unbiased estimator of the entropy gradient. We apply our method to a suite of tasks and empirically validate that AR-DAE provides accurate and reliable gradient signal to maximize entropy.

Acknowledgments

We would like to thank Guillaume Alain for an insightful discussion on denoising autoencoders. Special thanks to people who have provided their feedback and advice during discussion, including Bogdan Mazoure and Thang Doan for sharing the code on the RL experiment; to Joseph Paul Cohen for helping optimize the allocation of computational resources. We thank CIFAR, NSERC and PROMPT for their support of this work.

References

Agakov, F. V. and Barber, D. An auxiliary variational method. In *ICONIP*, 2004.

Alain, G. and Bengio, Y. What regularized auto-encoders learn from the data-generating distribution. *J. Mach. Learn. Res.*, 2014.

Balaji, Y., Hassani, H., Chellappa, R., and Feizi, S. Entropic gans meet vaes: A statistical approach to compute sample likelihoods in gans. In *ICML*, 2019.

Bertsekas, D. Nonlinear programming. 3rd edn. massachussets: Athena scientific, 2016.

Bigdeli, S. A., Lin, G., Portenier, T., Dunbar, L. A., and Zwicker, M. Learning generative models using denoising density estimators. *arXiv preprint arXiv:2001.02728*, 2020.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Burda, Y., Grosse, R. B., and Salakhutdinov, R. Importance weighted autoencoders. In *ICLR*, 2016.

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. Variational lossy autoencoder. In *ICLR*, 2016.

Dieng, A. B., Ruiz, F. J., Blei, D. M., and Titsias, M. K. Prescribed generative adversarial networks. *arXiv preprint arXiv:1910.04302*, 2019.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In *ICML*, 2016.

Durrett, R. *Probability: theory and examples*, volume 49. Cambridge university press, 2019.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *ICML*, 2018.

Geyer, C. J. Reweighting monte carlo mixtures. Technical report, University of Minnesota, 1991.

Ha, D., Dai, A. M., and Le, Q. V. Hypernetworks. In *ICLR*, 2017.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In *ICML*, 2017.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018.

Hasselt, H. V. Double q-learning. In *NIPS*, 2010.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Huang, C., Krueger, D., Lacoste, A., and Courville, A. C. Neural autoregressive flows. In *ICML*, 2018.

Huszár, F. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709, 2005.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *ICLR*, 2014.

Kingma, D. P., Salimans, T., Józefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improving variational autoencoders with inverse autoregressive flow. In *NIPS*, 2016.

Kumar, A., Poole, B., and Murphy, K. Regularized autoencoders via relaxed injective probability flow. *arXiv preprint arXiv:2002.08927*, 2020.

Larochelle, H. and Murray, I. The neural autoregressive distribution estimator. In *AISTATS*, 2011.

Li, Y. and Turner, R. E. Gradient estimators for implicit models. In *ICLR*, 2018.

Li, Y., Turner, R. E., and Liu, Q. Approximate inference with amortised mcmc. *arXiv preprint arXiv:1702.08343*, 2017.

Loaiza-Ganem, G., Gao, Y., and Cunningham, J. P. Maximum entropy flow networks. *arXiv preprint arXiv:1701.03504*, 2017.

Maaløe, L., Sønderby, C. K., Sønderby, S. K., and Winther, O. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

Mazoure, B., Doan, T., Durand, A., Hjelm, R. D., and Pineau, J. Leveraging exploration in off-policy algorithms via normalizing flows. *arXiv preprint arXiv:1905.06893*, 2019.

Mescheder, L. M., Nowozin, S., and Geiger, A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In *ICML*, 2017.

Minka, T. et al. Divergence measures and message passing. Technical report, Technical report, Microsoft Research, 2005.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Mohamed, S. and Lakshminarayanan, B. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

Odena, A., Buckman, J., Olsson, C., Brown, T. B., Olah, C., Raffel, C., and Goodfellow, I. Is generator conditioning causally related to gan performance? *arXiv preprint arXiv:1802.08768*, 2018.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

Polyak, B. T. and Juditsky, A. B. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

Ranganath, R., Tran, D., and Blei, D. Hierarchical variational models. In *ICML*, 2016.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. In *ICML*, 2015.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014a.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014b.

Roeder, G., Wu, Y., and Duvenaud, D. K. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *NIPS*, 2017.

Saremi, S. and Hyvarinen, A. Neural empirical bayes. *Journal of Machine Learning Research*, 20:1–23, 2019.

Saremi, S., Mehrjou, A., Schölkopf, B., and Hyvärinen, A. Deep energy estimator networks. *arXiv preprint arXiv:1805.08306*, 2018.

Savitzky, A. and Golay, M. J. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8):1627–1639, 1964.

Shalev-Shwartz, S., Shamir, O., and Shammah, S. Failures of gradient-based deep learning. In *ICML*, 2017.

Shi, J., Sun, S., and Zhu, J. A spectral approach to gradient estimation for implicit distributions. In *ICML*, 2018.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.

Song, Y., Garg, S., Shi, J., and Ermon, S. Sliced score matching: A scalable approach to density and score estimation. In *UAI*, 2019.

Sutton, R. S., Barto, A. G., et al. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge, 1998.

Tomczak, J. and Welling, M. Vae with a vampprior. In *AISTATS*, 2018.

Tran, D., Ranganath, R., and Blei, D. Hierarchical implicit models and likelihood-free variational inference. In *NIPS*, 2017.

Vincent, P. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P. Extracting and composing robust features with denoising autoencoders. In *ICML*, 2008.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

Wen, L., Zhou, Y., He, L., Zhou, M., and Xu, Z. Mutual information gradient estimation for representation learning. In *ICLR*, 2020.

Ziebart, B. D. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. 2010.