# Learning Generative Samplers using Relaxed Injective Flow

**Abhishek Kumar** [1] **Ben Poole** [1] **Kevin Murphy** [1]

## Abstract

Invertible flow-based generative models have been shown to be appealing as data samplers, while allowing for tractable likelihood computation and inference. However, the invertibility requirement results in latents that are of same dimensions as data, undermining the value of inference, particularly in the context of *representation* learning. The invertibility requirement also imposes significant memory and compute costs on these models, making it more challenging to scale them using reasonable resources, compared to other classes of generative models such as Variational Autoencoders (VAEs) or Generative Adversarial Networks (GANs). We propose a generative model based on probability flows that does away with the bijectivity requirement on the model and only assumes injectivity, with the goal of learning a sampling model for data. We empirically demonstrate the potential of the proposed model as a data sampler.

## 1. Introduction

Invertible flow-based generative models (Dinh et al., 2016; Kingma & Dhariwal, 2018) have recently gained traction due to several desirable properties: (i) exact log-likelihood calculation (unlike VAEs that maximize a lower bound (Kingma & Welling, 2013; Rezende et al., 2014)), (ii) exact inference of latent variables, (iii) good sample quality relative to VAEs, and (iv) fast sampling unlike autoregressive models (van den Oord et al., 2016).

However, invertible flow models require invertibility on the full ambient space, resulting in a latent space with the same dimensionality as the ambient data dimension. This results in larger models with higher memory and computational costs that are more difficult to scale than VAE and GAN counterparts with lower-dimensional latent spaces (Goodfellow et al., 2014; Kingma & Dhariwal, 2018). This lack of dimensionality reduction makes it difficult to capture high-level generative factors directly in latent dimensions, a property that is often argued to be desirable for generative models (Higgins et al., 2017; Narayanaswamy et al., 2017; Kumar et al., 2018; Kim & Mnih, 2018; Chen et al., 2018).

In this work we propose a generative model based on probability flows that relaxes the bijectivity requirement. The model $g$ maps low dimensional latents $z \in Z = \mathbb{R}^d$ to samples in the image of $g$, residing in much higher dimensional ambient space $\mathbb{R}^D$. A simple probability distribution in the latent space (such as standard normal) is pushed forward by the mapping $g$ to induce a distribution on the image $g(Z)$. By taking the mapping $g$ to be one-to-one or injective and differentiable, we use a change of variables theorem to obtain closed form for the distribution over $g(Z)$. We further lower bound the obtained log-likelihood and use a stochastic approximation to obtain a tractable objective amenable to stochastic first-order optimization. However, relaxing the bijectivity requirement loses the ability to provide likelihoods for data points that lie off the image $g(Z)$, preventing evaluation of models on test points with log-likelihood. Thus our goal in this work is to use the derived flow based objective to learn a sampling mechanism for the training data that always generates samples from the image $g(Z)$. This is in contrast to VAEs where generated samples can lie off the image $g(Z)$ due to the presence of an additional distribution at the output of the decoder (e.g., Gaussian).

Our final objective, although derived from probability flows, resembles a regularized autoencoder with an additional prior log probability term and a annealing on the reconstruction loss that increases its weight over time. For certain approximations of the intractable log-det-Jacobian term, our objective matches a regularized autoencoder objective suggested by (Ghosh et al., 2019), which recently demonstrated competitive sample quality with VAEs when a Gaussian mixture prior is fit to the latent space after training. We evaluate relaxed injective flows on MNIST, where we observe better FID scores with our model compared to VAEs and regularized autoencoder. Our results demonstrate that these models provide an efficient and tractable mechanism for training efficient neural samplers with compressed latent spaces.

[1]Google Research, Brain Team. Correspondence to: Abhishek Kumar <abhishk@google.com>, Ben Poole <pooleb@google.com>, Kevin Murphy <kpmurphy@google.com>.

First workshop on *Invertible Neural Networks and Normalizing Flows* (ICML 2019), Long Beach, CA, USA

## 2. Formulation

Let $g : Z \to X$ be the generator mapping from latents to data, assumed to be differentiable everywhere. If $g$ is a bijection with $h = g^{-1} : X \to Z$, then we can write the distribution induced over $X$ in terms of a distribution over $Z$ using change of variables formula:

$$\begin{aligned} \ln p_x(x) &= \ln p_z(h(x)) + \ln \left|\det J_h(x)\right|, \\ &= \ln p_z(h(x)) - \ln \left|\det J_g(h(x))\right| \end{aligned} \tag{1}$$

where $J_h(x)$ and $J_g(z)$ are the Jacobians of $h$ and $g$ at $x$ and $z$, respectively. Invertible flow models (Dinh et al., 2016; Kingma & Dhariwal, 2018) optimize (1) to learn a generative model of the data. The inference in these models comes for free as the inverse mapping can be tractably computed by making a careful choice on the architecture while maintaining bijectivity of $g$.

We are interested in developing probability flow based models for the setting when the dimensionality of the latent space $Z$ is much lower than the data dimensions, i.e., $Z = \mathbb{R}^d$ and $X = \mathbb{R}^D$, s.t. $D \gg d$. However we still assume that the intrinsic dimensionality of the data is much lower than the ambient dimensionality and is not greater than $d$. We can obtain a change of variables formula for this setting by looking at how an infinitesimal volume element $dz$ at $z \in Z$ changes by the mapping $g$. The theory of integration on manifolds tell us that the mapping $g$ transforms an infinitesimal volume element $dz$ at $z \in Z$ to $(J_g(z)^\top J_g(z))^{1/2} dz$ (Boothby, 1986). If we assume $g$ is an injective function and thus invertible when seen as a mapping $g : Z \to g(Z)$, we can write the distribution over $g(Z)$ as

$$\begin{aligned} \ln p_x(x) = \ln p_z(z) - \frac{1}{2} \ln \left|\det J_g(z)^\top J_g(z)\right|, \\ \text{s.t. } x = g(z). \end{aligned} \tag{2}$$

In order to avoid solving an inverse problem for every $x$ in our data, we assume the existence of an encoder $h : X \to Z$ such that $g(h(x)) = x$ for every $x \in g(Z)$. This gives

$$\begin{aligned} \ln p_x(x) = \ln p_z(h(x)) - \frac{1}{2} \ln \left|\det J_g(h(x))^\top J_g(h(x))\right|, \\ \text{s.t. } x = g(h(x)). \end{aligned} \tag{3}$$

Optimizing the $\ln \left|\det J_g(h(x))^\top J_g(h(x))\right|$ term is computationally challenging and we propose to lower bound the log likelihood in Eq. (3) in order to obtain a tractable objective. Assuming $J_g(h(x))$ admits a singular value decomposition with positive singular values $\{s_i\}_{i=1}^d$, we have

$$\begin{aligned} \ln p(x) &= \ln p(h(x)) - \frac{1}{2} \ln \det[J_g(h(x))^\top J_g(h(x))] \\ &= \ln p(h(x)) - \frac{1}{2} \text{tr}(\ln[J_g(h(x))^\top J_g(h(x))]) \\ &= \ln p(h(x)) - \frac{1}{2} \sum \ln s_i^2 \\ &\geq \ln p(h(x)) - \frac{d}{2} \ln \frac{1}{d} \sum s_i^2 \\ &= \ln p(h(x)) - \frac{d}{2} \ln \frac{1}{d} \text{tr}(J_g(h(x))^\top J_g(h(x))) \\ &= \ln p(h(x)) - \frac{d}{2} \ln \frac{1}{d} \mathbb{E}_v (v^\top J_g(h(x))^\top J_g(h(x)) v), \\ &\text{s.t. } x = g(h(x)). \end{aligned} \tag{4}$$

where $v \sim N(0, I)$. The inequality above is a result of using Jensen's inequality for log (*i.e.*, $\mathbb{E}_x \ln f(x) \leq \ln \mathbb{E}_x f(x)$). The inequality is tight when all the singular values of $J_g(h(x))$ are equal. To avoid explicitly materializing the full Jacobian, we have used the Hutchinson trace estimator (Hutchinson, 1990): $tr(A) = \mathbb{E}_v tr(Avv^\top) = \mathbb{E}_v v^\top Av$ for any random vector $v$ s.t. $\mathbb{E}_v vv^\top = I$.

This is a constrained optimization problem which can be solved using either penalty method or augmented Lagrangian method, which will convert it into an unconstrained objective. We use the penalty method for simplicity, obtaining the objective:

$$\begin{aligned} \ln p_z(h(x)) - \frac{d}{2} \ln \frac{1}{d} \mathbb{E}_v (v^\top J_g(h(x))^\top J_g(h(x)) v) \\ -\mu \|x - g(h(x))\|_2^2, \end{aligned} \tag{5}$$

where $\mu$ is a positive real that is increased as the optimization progresses (Bertsekas, 2016). We further employ stochastic approximation for the expectation inside log in the second term in Eq. (5). This gives us

$$\ln p_z(h(x)) - \frac{d}{2} \ln \sum_{i=1}^{k} \|J_g(h(x)) v_i\|_2^2 - \mu \|x - g(h(x))\|_2^2, \tag{6}$$

with $v_i \sim N(0, I)$ and ignoring the constant terms. Monte Carlo estimation used for the term $\ln \mathbb{E}_v \|J_g(h(x))v\|^2$ in (6) results in a biased estimator. The expectation of this estimator is lower than $\ln \mathbb{E}_v \|J_g(h(x))v\|^2$ which may not result in a lower bound on the log liklihood. Similar issue arises in some earlier works that try to do Monte Carlo estimation for $\ln \mathbb{E}_x f(x)$ (Li & Turner, 2016; Rhodes & Gutmann, 2018). In practice, we do not find it to be an issue. We also use finite difference approximation for estimating the norm of $J_g(h(x))v$ as

$$\|J_g(h(x))v\|_2^2 \approx \frac{\|g(z + \epsilon v) - g(z)\|^2}{\epsilon^2}, \tag{7}$$

for a sufficiently small $\epsilon > 0$. As $g$ is taken to be injective, we need to constrain all singular values of $J_g(h(x))$ to be positive. Instead of directly enforcing this which can be computationally challenging, we simply enforce that $v^\top (J_g(h(x))^\top J_g(h(x)))v$ is greater than a threshold $\eta$ for all $v$ with $\|v\| = 1$. Similar approach is used by Odena et al. (2018). Our final minimization objective for $k = 1$ is

$$\begin{aligned}\min_{g,h} \; & -\ln p_z(h(x)) + \mu \|x - g(h(x))\|_2^2 \\ & + \frac{d}{2} \ln \max\left( \frac{\|g(z+\epsilon v) - g(z)\|^2}{\epsilon^2}, \eta^2 \|v\|^2 \right) \\ & + \mu_{in} \left[ \min\left( \frac{\|g(z+\epsilon v) - g(z)\|}{\epsilon \|v\|} - \eta, 0 \right) \right]^2 \end{aligned} \tag{8}$$

where $v \sim N(0, I)$ and $\mu_{in}$ is also a positive penalty parameter which is increased as the optimization proceeds.

We obtain another tractable lower bound by using the inequality based on concavity of log: $\ln x \leq \frac{x}{\lambda} + \ln \lambda - 1$, for all $x > 0$ and $\lambda > 0$, as follows:

$$\begin{aligned} \ln p(x) &= \ln p(h(x)) - \frac{1}{2} \ln \det[J_g(h(x))^\top J_g(h(x))] \\ &\geq \ln p(h(x)) - \frac{1}{2} \sum_i \left( \frac{s_i^2}{\lambda} + \ln \lambda - 1 \right) \\ &= \ln p(h(x)) - \frac{1}{2\lambda} \|J_g(h(x))\|_F^2 - \frac{d}{2} \ln \lambda + \frac{d}{2}, \\ &\qquad \text{s.t. } x = g(h(x)). \end{aligned} \tag{9}$$

The inequality above is tight when $\lambda = s_i^2, \forall i$. Hence both inequalities (in (4), and (9)) are tight under the same condition of all singular values of the Jacobian being equal. In the general case of unequal singular values, the parameter $\lambda$ can be tuned to improve the tightness of the bound. It is also possible to maximize the lower bound in (9) w.r.t. $\lambda$, turning the overall optimization into a saddle point problem (*i.e.*, $\max_{h,g} \min_\lambda$), however, we take $\lambda$ to be a fixed hyperparameter (which can potentially be tuned) for the sake of simplicity. Note that it is also possible to use separate $\lambda_i$ corresponding to each $s_i$ to improve upon the tightness of the bound, however this will lead to a more complex optimization problem (whereas, taking $\lambda_i = \lambda$ leads to an optimization friendly Frobenius norm of the Jacobian in the objective). Following similar steps of forming an unconstrained objective using the penalty method and using Monte-Carlo estimation for $\|J_g(h(x))\|_F^2$, we obtain the following minimization objective corresponding to the lower bound of (9):

$$\begin{aligned}\min_{g,h} \; & -\ln p_z(h(x)) + \mu \|x - g(h(x))\|_2^2 \\ & + \frac{1}{2\lambda} \left[ \max\left( \frac{\|g(z+\epsilon v) - g(z)\|^2}{\epsilon^2}, \eta^2 \|v\|^2 \right) \right] \\ & + \mu_{in} \left[ \min\left( \frac{\|g(z+\epsilon v) - g(z)\|}{\epsilon \|v\|} - \eta, 0 \right) \right]^2 \end{aligned} \tag{10}$$

where $\lambda$ is a fixed hyperparameter (which is not optimized over but can be tuned as discussed earlier). We optimize the objectives (8) and (10) with respect to parameters of both generator $g$ and encoder $h$.

## 3. Related Work

Our work is similar in spirit to the recent work of Ghosh et al. (2019); van den Oord et al. (2017); Dai & Wipf (2019), that find regularized autoencoders paired with a learned prior produces high-quality samples. Our work provides another perspective on the regularized autoencoder (RAE) objective in (Ghosh et al., 2019), wherein the regularization terms arise naturally from approximating the log-likelihood objective of the injective probability flow. Ghosh et al. (2019) motivate the RAE objective by considering constant posterior-variance VAEs and connecting stochasticity at the decoder's input (arising by sampling from $q(z|x)$) to *smoothness* of the decoder.

Regularized autoencoders have been widely studied in earlier works as well (Rifai et al., 2011; Alain & Bengio, 2014; Poole et al., 2014). Contractive autoencoders (Rifai et al., 2011) also penalize the Frobenius norm of the Jacobian, however the penalty is on the *encoder* Jacobian, which is different from our penalty on the *decoder* Jacobian. Most of these prior works on RAEs has been on improving the quality of the encoder for downstream tasks, where we are primarily interested in the quality of the generator for producing samples. Recent work has turned to regularizing autoencoders for sample quality as well, for example improving interpolation quality using an adversarial training objective (Berthelot et al., 2018).

Krusinga et al. (2019) recently used Eq. (2) to get density estimates for trained GANs. However as we noted earlier, these density estimates are by nature undefined for unseen real example which may lie off the manifold. On the other hand, we develop this probability flow further to obtain an objective that can be optimized to learn a generative sampler of the data.

Several earlier works have also used spectral regularizers in training generative models. Miyato et al. (2018) encourage Lipschitz smoothness of the GAN discriminator by normalizing the spectral norm of each layer. Odena et al. (2018)

study the spectral properties of the Jacobian of the generator and its correlation with the quality of generated samples. They empirically observe that regularizing the condition number of the Jacobian leads to more stable training and improved generative model. On the other hand, our Jacobian regularizer arises naturally from considering the probability flow from latent space to the observation space.

## 4. Experiments

We test the proposed model on MNIST dataset. Our final objective, although obtained by developing an injective flow and lower bounding its log likelihood, has resemblance with recently proposed regularized autoencoders (Ghosh et al., 2019) which arise as natural models for comparison. We compare our model with **(i) AE**: Vanilla autoencoder trained with $\ell_2$ reconstruction loss. **(ii) AE+L2**: Autoencoder with an additional $\ell_2$-norm penalty on the decoder parameters (weight decay), and **(iii) VAE** (Kingma & Welling, 2013) with a Gaussian observation model at the decoder's output $N(0, I_D)$.

We use five layer convolutional neural net based architecture for both encoder and decoder, each having five layers of convolutions or transposed convolutions, respectively. Strides and kernel-size in the convolutional filters stay same for all the models. We use *elu* activation in both encoder and decoder, and also use batch normalization. Latent dimensionality is taken to be 32 for MNIST.

We use $p(z) = N(0, \sigma^2 I)$ for the prior distribution. Our log-Frobenius norm objective (8) (referred as InjFlow$^{\ell n}$) has four hyperparameters: (i) variance $\sigma^2$ of prior on latent space, which determines the weight on the term penalizing the norm of the encodings $\|h(x)\|_2^2$, (ii) penalty coefficient $\mu$ on the reconstruction loss, (iii) penalty coefficient $\mu_{in}$ on the injectivity loss term, and (iv) singular value threshold used in the injectivity term $\eta$. We use $\eta = 0.1$ in all our experiments. Both penalty coefficients $\mu$ and $\mu_{in}$ are initialized to be 1 at the beginning of optimization and are increased with each minibatch iteration $i$ as $1 + \frac{i^\nu}{1000}$, where $\nu$ is searched over $\{1, 1.3\}$. The weight on the prior term $\|h(x)\|^2$ is searched over $\{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$. Our objective (10) (referred as InjFlow) has an additional hyperparameter $\lambda$ that determines the weight on Frobenius norm regularization term, which we fix to 1 in all our experiments. For AE+L2, the hyperparameter for the $\ell_2$ regularization term is searched over the set $\{0.001, 0.01, 0.1, 0.5, 1\}$. We train all models using Adam optimizer (Kingma & Ba, 2014) with batch size of 128 and a fixed learning rate of 0.001. All models are trained for 100k minibatch iterations. Following several earlier works, including (Tolstikhin et al., 2017; Dai & Wipf, 2019; Ghosh et al., 2019), we use FID score (Heusel et al., 2017) as a metric for sample quality.

*Table 1.* FID scores (lower is better). *Reconstructions:* score for reconstructed test data, *Gaussian:* score for decoded samples from a Gaussian prior with full covariance fit to encoded training samples, *GMM:* score for decoded samples from a GMM fit to encoded training samples. InjFlow and InjFlow$^{\ell n}$ are the models obtained from the objectives (10) and (8), respectively (with superscript $\ell n$ denoting the presence of log with the Frobenius term in (4) and (8).

| | Reconstructions | Samples | |
|---|---|---|---|
| | | Gaussian | GMM |
| VAE | 65.10 | 57.04 | 62.08 |
| Autoencoder | 8.69 | 43.40 | 12.14 |
| InjFlow$^{\ell n}$ | 7.40 | 35.96 | 9.93 |
| InjFlow | 6.0 | 42.65 | 11.43 |

Table 1 shows the FID scores for reconstructed test samples as well as for samples from the decoder. Following (Dai & Wipf, 2019; Ghosh et al., 2019) we fit a distribution on the encoded training points after training the model. Dai & Wipf (2019) train another VAE on the encoded training data but we fit a Gaussian and a mixture of 10 Gaussians similar to (Ghosh et al., 2019).

For all models, we report the best FID score obtained using decoder sampling from a post-fit GMM in the latent space. We then report all the other scores (i.e., samples from $N(0, I)$, samples from post-fit Gaussian, test reconstructions) for the same decoder that gives the best GMM samples FID score. This enables us to assess models in term of their best possible sample generation ability. The proposed injective flow model yields better FID scores than the baseline models.

Figures 1 and 2 show the random decoded samples from a Gaussian and a GMM fit to encoded training samples. Test reconstructions for randomly selected data points are also shown.

*Figure 1.* Samples from the autoencoder. *Top and middle:* samples from Gaussian and GMM distributions post-fit to encoded training data, respectively. *Bottom:* reconstructed test samples.

## 5. Discussion

We proposed a probability flow based generative model that works with injective generator mapping, relaxing the bijec-

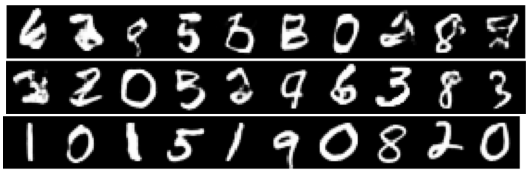

*Figure 2.* Samples from the relaxed injective flow model. *Top and middle:* samples from Gaussian and GMM distributions post-fit to encoded training data, respectively. *Bottom:* reconstructed test samples.

tivity requirement. We use a change of variables formula to derive an optimization objective for learning the generator and encoder, where a smoothness regularizer on the generator naturally arises from the probability flow, along with an additional penalty term for injectivity constraint. Note that the learned generative model $g$ may violate injectivity or perfect reconstruction constraints on test or even on training points. However, our goal in this work is to use this objective to only learn a sampling mechanism for the data.

Relaxing the bijectivity constraint makes use lose some nice properties of invertible flow based generative models, such as tractable likelihood and inference for unseen data. A possible approach to recover these aspects could be to do define a background probability model over the full ambient space $X$ and work with a mixture of foreground distribution over $g(Z)$ coming from probability flow and this background distribution. Investigation of this is an interesting future direction.

## References


Alain, G. and Bengio, Y. What regularized auto-encoders learn from the data-generating distribution. *Journal of Machine Learning Research*, 15(1):3563–3593, 2014.

Berthelot, D., Raffel, C., Roy, A., and Goodfellow, I. Understanding and improving interpolation in autoencoders via an adversarial regularizer. *arXiv preprint arXiv:1807.07543*, 2018.

Bertsekas, D. P. Nonlinear programming: 3rd edition. 2016.

Boothby, W. M. *An introduction to differentiable manifolds and Riemannian geometry*, volume 120. Academic press, 1986.

Chen, T. Q., Li, X., Grosse, R. B., and Duvenaud, D. K. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pp. 2610–2620, 2018.

Dai, B. and Wipf, D. Diagnosing and enhancing vae models. *arXiv preprint arXiv:1903.05789*, 2019.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Ghosh, P., Sajjadi, M. S. M., Vergari, A., Black, M., and Schlkopf, B. From variational to deterministic autoencoders, 2019.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

Hutchinson, M. F. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 19(2):433–450, 1990.

Kim, H. and Mnih, A. Disentangling by factorising. *arXiv preprint arXiv:1802.05983*, 2018.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pp. 10215–10224, 2018.

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

Krusinga, R., Shah, S., Zwicker, M., Goldstein, T., and Jacobs, D. Understanding the (un) interpretability of natural image distributions using generative models. *arXiv preprint arXiv:1901.01499*, 2019.

Kumar, A., Sattigeri, P., and Balakrishnan, A. Variational inference of disentangled latent concepts from unlabeled observations. In *ICLR*, 2018.

Li, Y. and Turner, R. E. Rényi divergence variational inference. In *Advances in Neural Information Processing Systems*, pp. 1073–1081, 2016.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Narayanaswamy, S., Paige, T. B., Van de Meent, J.-W., Desmaison, A., Goodman, N., Kohli, P., Wood, F., and Torr, P. Learning disentangled representations with semi-supervised deep generative models. In *Advances in Neural Information Processing Systems*, pp. 5925–5935, 2017.

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.

Poole, B., Sohl-Dickstein, J., and Ganguli, S. Analyzing noise in autoencoders and deep networks, 2014.

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

Rhodes, B. and Gutmann, M. Variational noise-contrastive estimation. *arXiv preprint arXiv:1810.08010*, 2018.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pp. 833–840.

Omnipress, 2011.

Tolstikhin, I., Bousquet, O., Gelly, S., and Schoelkopf, B. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.

van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.

van den Oord, A., Vinyals, O., et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pp. 6306–6315, 2017.