International Journal of Computer Vision (2020) 128:2731–2743
https://doi.org/10.1007/s11263-020-01360-9

GADE: A Generative Adversarial Approach to Density Estimation and its Applications

M. Ehsan Abbasnejad[1] · Javen Shi[1] · Anton van den Hengel[1] · Lingqiao Liu[1]

Received: 29 April 2019 / Accepted: 15 July 2020 / Published online: 19 August 2020

Abstract
Density estimation is a challenging unsupervised learning problem. Current maximum likelihood approaches for density estimation are either restrictive or incapable of producing high-quality samples. On the other hand, likelihood-free models such as generative adversarial networks, produce sharp samples without a density model. The lack of a density estimate limits the applications to which the sampled data can be put, however. We propose a *generative adversarial density estimator* (GADE), a density estimation approach that bridges the gap between the two. Allowing for a prior on the parameters of the model, we extend our density estimator to a Bayesian model where we can leverage the predictive variance to measure our confidence in the likelihood. Our experiments on challenging applications such as visual dialog or autonomous driving where the density and the confidence in predictions are crucial shows the effectiveness of our approach.

Keywords Generative models · GANs · Flow-based generative models · Deep learning

1 Introduction

Generative modelling is amongst the longest-standing problems in machine learning, and one that has been drawing increasing attention. This is at least partly due to the shortcomings of the predominant discriminative deep learning-based models. These shortcomings include the failure to generalise, a lack of robustness to data distribution changes, and the need for large volumes of training data.

Deep generative models have been successful in addressing some of these shortcomings. In particular, deep maximum likelihood models such as deep Boltzmann machines (Salakhutdinov and Hinton 2009), variational autoencoders (VAEs) (Kingma and Welling 2014), autoregressive models (Gregor et al. 2014; Oord et al. 2016), real non-volume preserving transformations (Dinh et al. 2016) etc.have demonstrated an impressive ability to model complex densities.

Likelihood-free approaches (Gutmann et al. 2018; Goodfellow et al. 2014) such as generative adversarial networks (GANs) (Goodfellow et al. 2014) have outperformed previous deep generative models in their ability to model complex distributions. In image-based problems they have shown a particular ability to consistently generate sharp and realistic looking samples (Karras et al. 2017; Zhang et al. 2017; Nguyen et al. 2017). GANs are one of the *implicit* generative models wherein density is not explicitly defined (Diggle and Gratton 1984). Unfortunately, GANs are incapable of evaluating densities by-design.

Maximum likelihood models have the advantage that they are able to represent probability density explicitly, but either perform poorly in terms of the quality of samples they generate or are relatively inefficient to train.

As modern datasets are typically large, high-dimensional and highly structured, the challenge is building models that are powerful enough to capture the underlying complexity yet still efficient to train in estimating the density of instances.

Density estimates are essential in a wide range of practical problems in Computer Vision, particularly when likelihoods over the generated samples are critical. This occurs, for

Communicated by Jun-Yan Zhu, Hongsheng Li, Eli Shechtman, Ming-Yu Liu, Jan Kautz, Antonio Torralba.

✉ M. Ehsan Abbasnejad
ehsan.abbasnejad@adelaide.edu.au

Javen Shi
javen.shi@adelaide.edu.au

Anton van den Hengel
anton.vandenhengel@adelaide.edu.au

Lingqiao Liu
lingqiao.liu@adelaide.edu.au

[1] Australian Institute for Machine Learning, The University of Adelaide, Adelaide, Australia

example, when it is important to both explore and optimise over a search space, when confidence estimates about a hypothesis are required, or when control over the level of generalisation is important. The typical sample quality metrics are inadequate since the generative model may simply memoraize the data or miss important modes (Theis et al. 2015). Moreover, in an application such as one where the virtual agent is engaged in a dialog with the user about an image, the estimate of how likely an answer is in addition to the agent's confidence improves both dialogue fluidity and method performance. Alternatively, density estimates such as autoregressive models (Gregor et al. 2014), flow-based methods (Dinh et al. 2014, 2016) or non-parameteric methods such as kernel density estimation (KDE) (Goodfellow et al. 2014) are either overly computationally demanding or rely on heavy engineering of the neural networks involved.

In this paper, we introduce a *Generative Adversarial Density Estimator* that is both easy to train and expressive enough to model high-dimensional data. In particular, we bridge the gap between the maximum likelihood and likelihood-free models by explicitly modeling the likelihood while using adversarial samples to compute the normalizer. As a by-product of this model, we show the local properties of the density function are captured and allow for diverse samples to be generated. Further, we show our approach is capable of modeling the likelihood of complex data, such as images, from which realistic looking samples can be taken.

In addition, our approach easily extends to Bayesian estimation where the distribution of the parameters are incorporated in the generative model. This allows us to compute the predictive variance which uncovers the "uncertainty" in the prediction. This uncertainty can be used in applications such as Visual Dialog (Das et al. 2017) where an agent is trained to respond to questions when confident or reply "I don't know" for uncertain answers. This is essential if such agents are to be of practical utility.

Moreover, we employ our approach in Bayesian imitation learning (Price and Boutilier 2003; Ramachandran and Amir 2007) where uncertainty is crucial in determining the actions to take. For instance, in autonomous driving where uncertain decisions are costly and lead to human casualties, employing our approach is highly desirable. Hence, we demonstrate an application of our approach to autonomous driving using the TORCS driving simulator (Espié et al. 2000). This paper makes the following contributions:

1. We propose a generative adversarial density estimator network able to perform efficient sampling as well as likelihood evaluation without the need for an explicit invertible generator.
2. We propose a learning objective, based on sound theoretical grounds, for our adversarial density estimator that attains good log-likelihoods and generates high-quality samples on multiple datasets as well as estimating uncertainty in a vision-and-language problem.
3. Our approach is naturally extended to a Bayesian setting. In particular we investigate a conjugate prior and an efficient gradient based Monte Carlo method to estimate the posterior.
4. We propose a method to efficiently regularize the Jacobian of the generator function to evaluate the likelihoods of the model.
5. We provide a theoretical apparatus for future research in density estimation, in particular, utilizing adversarial training.
6. We show how our adversarial density estimation approach is capable of improving imitation learning for the challenging task of autonomous driving.

2 Related Work

Deep Boltzmann machines (Salakhutdinov and Hinton 2009) represent one of the earlier approaches to maximum likelihood modelling, but due to their intractable nature, they are hard to train. Recently, autoregressive approaches (Oord et al. 2016; Salimans et al. 2017) have been used to model the distribution of each image pixel value directly. As such, the ordering of the dimensions can be critical to the training of the model. The sequential nature of these models significantly limits their computational efficiency. VAEs (Kingma and Welling 2014) have shown to be successful in modeling a latent variable from which instances from the distribution can be reconstructed. While these approaches have been very successful, image samples generated by VAEs are generally blurry and exhibit unrealistic artefacts.

Generative Adversarial Networks (GANs) (Goodfellow et al. 2014) avoid the maximum likelihood principle altogether. Instead, the generative network is associated with a discriminator network whose task is to distinguish between samples and real data. Successfully trained GAN models consistently outperform their counterparts in producing high-quality, sharp and realistic image samples (Wang et al. 2017; Karras et al. 2017). However, GANs cannot estimate the density function over the sampled data instances, even though approaches such as Wasserstein distance (WGAN) (Arjovsky et al. 2017; Gulrajani et al. 2017) and f-GAN (Nowozin et al. 2016) are similar to ours. Recently, (Rezende and Mohamed 2015); Grover et al. 2018) considered likelihood maximisation with a change of variable. However, these approaches require designing a function that is both invertible and that has an easily computable Jacobian. These designs typically produce low quality samples, or are inefficient.

The connection between GANs and actor-critic models in reinforcement learning (RL) has already been made (Pfau and Vinyals 2016). In addition, (Finn et al. 2016) pointed out the

connection between energy-based models, GANs and inverse RL that is closely related to our approach. However, the relation between the density and the entropy of the generator is not considered. The Bayesian approaches for inverse RL were introduced by Ramachandran and Amir (2007). GANs have also been used in imitation learning, to estimate the reward function (Li et al. 2017; Ho and Ermon 2016). Moreover, maximum entropy inverse RL (Ziebart et al. 2008) and guided policy search (Levine and Koltun 2013) are related to ours.

3 Generative Adversarial Density Estimator

3.1 Adversarial Formulation

We consider the problem of estimating the density for an observation sample $\mathbf{x}$ (e.g. an image) defined by

$$p(\mathbf{x}|\mathbf{w}) = \exp(\phi(\mathbf{w},\mathbf{x}) - A(\mathbf{w})),$$
$$A(\mathbf{w}) = \log\left(\int \exp(\phi(\mathbf{w},\mathbf{x}))d\mathbf{x}\right). \quad (1)$$

here $\phi(\mathbf{w},\mathbf{x})$ is the energy function for the Boltzmann distribution and the parameter $\mathbf{w}$ fully specifies the density function. We use a deep neural network with linear output to model this function, i.e. $\phi(\mathbf{w},\mathbf{x}) = \mathbf{w}_1^\top \phi'(\mathbf{w}_2,\mathbf{x})$, where $\mathbf{w} = \{\mathbf{w}_1, \mathbf{w}_2\}$ and feature vector $\phi'(\mathbf{w}_2,\mathbf{x})$ is the *sufficient statistics* that we learn. Alternatively ϕ can be seen as a function that maps the input $\mathbf{x}$ to a feature space where the density is easy to formulate. In addition, $A(\mathbf{w})$ is the *normalizer* that ensures $\int p(\mathbf{x}|\mathbf{w})d\mathbf{x} = 1$.

Given a dataset $\mathcal{X} = \{\mathbf{x}_1 \ldots, \mathbf{x}_n\}$ where samples are drawn i.i.d from the underlying distribution p, we are interested in finding parameter $\mathbf{w}$ in Eq. 1. To that end, we maximize the expected log-likelihood of these observations in the dataset with respect to this density model to obtain the parameter $\mathbf{w}$, that is, $\max_{\mathbf{w}} \mathbb{E}_p\left[\log p(\mathbf{x}|\mathbf{w})\right]$, where

$$\mathbb{E}_p\left[\log p(\mathbf{x}|\mathbf{w})\right] = \int \log(p(\mathbf{x}|\mathbf{w}))p(\mathbf{x}|\mathbf{w})d\mathbf{x}$$
$$= \mathbb{E}_p\left[\phi(\mathbf{w},\mathbf{x})\right] - A(\mathbf{w}) \quad (2)$$

Computing the normalizer $A(\mathbf{w})$ is intractable, and thus we resort to approximations. In particular, we utilize the variational principle to obtain an upper bound on $A(\mathbf{w})$ using an alternative distribution q parameterized by $\boldsymbol{\theta}$ as

$$A_q(\mathbf{w}) = \sup_q \Big\{ \overbrace{\int \phi(\mathbf{w},\mathbf{x})q(\mathbf{x}|\boldsymbol{\theta})d\mathbf{x}}^{\mathbb{E}_q[\phi(\mathbf{w},\mathbf{x})]} + \underbrace{\int -q(\mathbf{x}|\boldsymbol{\theta})\log\left(q(\mathbf{x}|\boldsymbol{\theta})\right)d\mathbf{x}}_{H_{\mathbf{x}}(q)} \Big\} \quad (3)$$

Note that direct maximization of $A_q(\mathbf{w})$ yields the optimal value when $p = q$, i.e. when q takes the same form as p. Classical variational inference methods (see e.g. (Kingma and Welling 2014)) define a class of distributions from which a q is selected (or rather parameter $\boldsymbol{\theta}$ is obtained). Here, motivated by the approach used in GANs, we specify q through a likelihood-free deterministic function which we use to generate samples (hence called generator). Assuming this generator function $\mathbf{x} = g_{\boldsymbol{\theta}_g}(\mathbf{z})$, $\mathbf{z} \sim p_z$ is invertible,[1] by the calculus of variable change (Ben-Israel 1999) $\mathbb{E}_q[\phi(\mathbf{w},\mathbf{x})]$ in Eq. 3 is:

$$\int \phi(\mathbf{w},\mathbf{x})q(\mathbf{x})d\mathbf{x} =$$
$$\int \phi(\mathbf{w}, g_{\boldsymbol{\theta}_g}(\mathbf{z}))p_z(g_{\boldsymbol{\theta}_g}^{-1}(\mathbf{x})|\boldsymbol{\theta}_z,\boldsymbol{\theta}_g)\left|\det\left(J_{\mathbf{x}}^\top J_{\mathbf{x}}\right)\right|^{1/2}d\mathbf{z}$$
$$\text{and } p_z(g_{\boldsymbol{\theta}_g}(\mathbf{x})^{-1}|\boldsymbol{\theta}_z,\boldsymbol{\theta}_g)\left|\det\left(J_{\mathbf{x}}^\top J_{\mathbf{x}}\right)\right|^{1/2}d\mathbf{z} = p_z(\mathbf{z}|\boldsymbol{\theta}_z)d\mathbf{z}, \quad (4)$$

where p_z is the prior distribution with parameter $\boldsymbol{\theta}_z$ and $J_{\mathbf{x}}$ is the Jacobian of $g_{\boldsymbol{\theta}_g}(\mathbf{x})^{-1}$. Here we use the convention that $\boldsymbol{\theta}$ is composed of two parts: $\boldsymbol{\theta}_g, \boldsymbol{\theta}_z$ for the parameters of the transformation function $g_{\boldsymbol{\theta}_g}(\cdot)$ and the distribution of $\mathbf{z}$, respectively. In the second line of Eq. 4, we specify that $g_{\boldsymbol{\theta}_g}$ preserves the volume with respect to $\phi(\mathbf{w},\mathbf{x})$. Note in practice we use $\phi(\mathbf{w}_2,\mathbf{x})$ as a surrogate for g^{-1} and regularize it to have a relatively constant determinant (by adding a regularization term for the gradients of $\phi(\mathbf{w}_2,\mathbf{x})$).

To compute the entropy of q in $H_{\mathbf{x}}(q)$ in Eq. 3, again using variable change, we have $\log p_z(\mathbf{z}) = \log q(g(\mathbf{z})) + \log(\left|\det\left(J_{\mathbf{z}}^\top J_{\mathbf{z}}\right)\right|^{1/2})$ and $q(g(\mathbf{z})) = q(\mathbf{x})$ for which the expectation yields,

$$H_{\mathbf{x}}(q) = \overbrace{H(p_z)}^{\text{entropy of prior noise}} + \overbrace{H_g}^{\text{local geometry of the transformed space}},$$
$$\text{where} \quad H_g = -\mathbb{E}_{p_z}\left[\log\left|\det(J_{\mathbf{z}}^\top J_{\mathbf{z}})\right|^{1/2}\right] \quad (5)$$

where $J_{\mathbf{z}}$ is the Jacobian of $g_{\boldsymbol{\theta}_g}$ (the generator). Intuitively, this equality states that the entropy with respect to the distribution of the generated samples is the sum of two terms (1) the entropy of the noise *prior* which captures the *information*

[1] This is merely for theoretical analysis. In practice, we don't need to explicitly define an invertible function.

present in the noise distribution (specified by parameter $\boldsymbol{\theta}_z$) and (2) the expected local behavior in the transformed space (specified by parameter $\boldsymbol{\theta}_g$). From the information geometric perspective, the latter term captures the volume of the geometric space that the transformation function $g_{\boldsymbol{\theta}_g}$ produces.

Remark 1 Explicitly constraining the generator to be invertible is infeasible, however, in practice we can use networks with inverted architecture to mediate this. Furthermore, when the matrix $J_{\mathbf{z}}^{\top} J_{\mathbf{z}}$ is full-rank, it indicates the mapping is one-to-one, i.e. there is an invertible function that maps samples from $\mathbf{z}$ to observation samples $\mathbf{x}$.

Since computing the log det term in Eq. 5 is computationally expensive, we resort to an approximation that is easier to optimize with stochastic gradient descent methods. We utilize the following Lemma[2]:

Lemma 2 *For a matrix $A \in \mathbb{R}^{d\times d}$ and $\mathbf{I}_d$ the identity matrix of size d where $\lambda_1(A)$ denotes the largest eigenvalue and $\lambda_1(A) < \alpha$ we have*

$$\log\det(A) = d\log(\alpha) + \sum_{k=1}^{\infty} \frac{(-1)^{k+1}\,\mathrm{tr}\left(\left(A/\alpha - \mathbf{I}_d\right)^k\right)}{k}. \tag{6}$$

Proof Refer to the Appendix. □

Employing Lemma 2, we are able to substitute the computation of the determinant for a more efficient alternative approximation. This approximation is arbitrarily close to the true value depending on the number of steps in the series and the value chosen for α.

Remark 3 One can use our approach in a flow-based model to generate samples and compute the density. This corresponds to when q estimates the true data distribution, which is a special case in our approach. However, it is generally harder to have one network achieve both tasks (generation and density estimation) as such we consider decoupling them.

3.2 Main Algorithm

Taking the derivative of the normalizer w.r.t. the parameter $\mathbf{w}_1$ in Eq. 1 yields the first moment of this distribution, i.e. $\mathbb{E}[\phi'(\mathbf{w}_2, \mathbf{x})] = \frac{\partial A(\mathbf{w})}{\partial \mathbf{w}_1}$. Furthermore, the second derivative is the covariance (and in this case Fisher information specifying the sensitivity of the parameter to the input). As such, it matches the moments from the true distribution and its estimate with respect to the sufficient statistics $\phi'(\mathbf{w}_2, \mathbf{x})$ when using maximum likelihood.

[2] A randomized version of this lemma is proposed in Boutsidis et al. (2017).

Putting Eqs. 1, 3, 4 and 5 together, we have a max-min problem to solve for two sets of parameters: $\mathbf{w}$ and $\boldsymbol{\theta}$. Finding the saddle point solution for this problem is challenging, as such, we employ an alternating optimization using stochastic gradient descent to update the parameters $\mathbf{w}$ and $\boldsymbol{\theta}$ iteratively, as

$$\begin{aligned}
\mathbf{w} &\leftarrow \mathbf{w} + \eta_w \nabla_{\mathbf{w}} \Big(\mathbb{E}_p\Big[\phi(\mathbf{w}, \mathbf{x})\Big] - \mathbb{E}_{p_z}\Big[\phi(\mathbf{w}, g_{\boldsymbol{\theta}_g}(\mathbf{z}))\Big] \Big) \\
\boldsymbol{\theta} &\leftarrow \boldsymbol{\theta} + \eta_g \nabla_{\boldsymbol{\theta}} \Big(\mathbb{E}_{p_z}\Big[\phi(\mathbf{w}, g_{\boldsymbol{\theta}_g}(\mathbf{z}))\Big] + H(p_z) + H_g \Big),
\end{aligned} \tag{7}$$

In general, we need to optimize $\mathbf{w}$ at each step until convergence using the $\boldsymbol{\theta}$ found in the previous iteration. In practice, we use a smaller learning rate to update $\boldsymbol{\theta}$ (or fewer rounds of update) to guarantee convergence (Konda and Tsitsiklis 2004).

The transformation function acts similar to the generator in GANs is mode-seeking and it is complemented by the ability of maximum likelihood to cover the space of $\mathcal{X}$. In particular, maximizing the entropy in the input noise and the output ensures a transformation function $g_{\boldsymbol{\theta}_g}$ that does not collapse.

Based on the definition we provided in Eq. 4, the ideal distribution for p_z is one that follows a similar distribution. In particular, $\mathbf{z}$ is more likely for the most frequent samples in the true distribution for which $\phi(\mathbf{w}, \mathbf{x})$ is the highest. However, as a prior we can choose a simple distribution such as a Gaussian whose entropy $H(p_z)$ can be analytically computed in closed-form (e.g. for Guassian $\log(\sigma\sqrt{2\pi e})$ for standard deviation σ).

Remark 4 It is interesting to note that if $g_{\boldsymbol{\theta}_g}(\cdot) = \phi'^{-1}(\cdot)$ then from Eq. 4, we have $\mathbb{E}_q[\mathbf{w}_1^{\top}\phi'(\mathbf{w}_2, \mathbf{x})]$ equals $\mathbb{E}_{p_z}[\mathbf{w}_1^{\top}\phi'(\mathbf{w}_2, g_{\boldsymbol{\theta}_g}(\mathbf{z}))] = \mathbb{E}_{p_z}[\mathbf{w}_1^{\top}\mathbf{z}]$, hence $\mathbb{E}_q[\phi'(\mathbf{w}_2, \mathbf{x})] = \mathbb{E}_{p_z}[\mathbf{z}]$ and $\mathbb{E}_p[\phi'(\mathbf{w}_2, \mathbf{x})] = \mathbb{E}_{p_z}[\mathbf{z}]$. This entails distribution of prior $\mathbf{z}$ should match that of ϕ in expectation. In practice, constraining $g_{\boldsymbol{\theta}_g}$ to be an inverse of ϕ (even in expectation) is too restrictive and hard to implement.[3] However, it is fruitful to know when constructing neural networks for this task, an inverse architecture for the networks learning $g_{\boldsymbol{\theta}_g}$ and ϕ has the best chance of success. Similarly, it is based on this argument and the entropy in Eq. 5 that it is better to learn the distribution of $\mathbf{z}$. However, it should be noted that since $g_{\boldsymbol{\theta}_g}$ is a deterministic function, changes to the distribution of $\mathbf{z}$ can negatively impact our generator or decrease the convergence speed of $g_{\boldsymbol{\theta}_g}$ network. In practice, we make sure the learning rate of $\boldsymbol{\theta}_z$ is smaller than $\boldsymbol{\theta}_g$, its counterpart for $g_{\boldsymbol{\theta}_g}$.

[3] We observed in practice that if the generator function g_z shares weights with that of ϕ (i.e. second layer of g_z with last layer of ϕ and so forth) the approach does not perform well. In addition, even if we constrain each layer of the g_z network to match in distribution to layers of ϕ the quality of the generated samples deteriorate.

Algorithm 1 Our density estimation algorithm.

Input $\mathcal{X}$, learning rates η_w, η_g, η_z, $\eta_z < \eta_g < \eta_w$, regularization parameter λ
while $\boldsymbol{\theta}$ is not converged **do**
  Sample uniformly $\mathbf{x}_1, \ldots, \mathbf{x}_n$ from $\mathcal{X}$
  Sample $\mathbf{z}'_1, \ldots, \mathbf{z}'_m \sim p_z(\mathbf{z}|\boldsymbol{\theta}_z)$
  $\rho'_{\mathbf{w}} = \nabla_{\mathbf{w}} \left[\frac{1}{n} \sum_i (\phi(\mathbf{w}, \mathbf{x}_i)) - \frac{1}{m} \sum_i (\phi(\mathbf{w}, g_{\boldsymbol{\theta}}(\mathbf{z}_i))) \right]$
  $R = -\|\nabla_{\mathbf{w}_2} \phi'(\mathbf{w}_2, \mathbf{x})\|^2$
  $\mathbf{w} = \mathbf{w} + \eta_w.\text{Adam}(\mathbf{w}, \rho'_{\mathbf{w}} + \lambda \nabla_{\mathbf{w}} R)$
  Sample $\mathbf{z}'_1, \ldots, \mathbf{z}'_m \sim p_z(\mathbf{z}|\boldsymbol{\theta}_z)$
  $\hat{H}_g = \frac{1}{2} \log\det(J_{\mathbf{z}}^\top J_{\mathbf{z}})$ ▷ Using Lemma 2
  $\rho'_{\boldsymbol{\theta}_{\{g,z\}}} = \nabla_{\boldsymbol{\theta}_{\{g,z\}}} \frac{1}{m} \sum_i \left[\phi(\mathbf{w}, g_{\boldsymbol{\theta}_{\{g,z\}}}(\mathbf{z}_i)) + \hat{H}_g \right]$
  $\boldsymbol{\theta}_g = \boldsymbol{\theta}_g + \eta_g.\text{Adam}(\boldsymbol{\theta}_g, \rho'_{\boldsymbol{\theta}_g})$
  $\boldsymbol{\theta}_z = \boldsymbol{\theta}_z + \eta_z.\text{Adam}(\boldsymbol{\theta}_z, \nabla_{\boldsymbol{\theta}_z} \left(H(p_z) + \rho'_{\boldsymbol{\theta}_z} \right))$
end while

To estimate the integrals in the normalizer $A_q(\mathbf{w})$ we employ Monte Carlo method using samples the generator produces and Lemma 2 to obtain $\hat{A}(\mathbf{w})$, i.e.

$$\hat{A}(\mathbf{w}) = \frac{1}{m} \sum_{j=1}^{m} \phi(\mathbf{w}, \mathbf{x}_j) + H(p_z) + \hat{H}_g$$
$$\text{where} \quad \mathbf{x}_j = g_{\boldsymbol{\theta}_g}(\mathbf{z}_j), \mathbf{z}_j \sim q(\mathbf{z}_j|\boldsymbol{\theta}_z), j = 1, \ldots, m \tag{8}$$

Here, $\hat{H}_g$ is the empirical estimate of H_g using samples $\mathbf{z}_j$ for the Jacobian of the transformation function $g_{\boldsymbol{\theta}_g}$.

The summary of the algorithm is shown in Algorithm 1 where we employ this alternating optimization. In our approach density estimator maximizes the log-likelihood of the given data with respect to the model in Eq. 1. Generated samples from the transformation function $g_{\boldsymbol{\theta}_g}$ are the byproduct of this maximization in the normalizer.

We note that our approach in maximizing the H_g in Eq. 5 resembles the ones in volume preserving or non-volume preserving generative models (Dinh et al. 2016, 2014) and normalized flows (Grover et al. 2018; Rezende and Mohamed 2015)) that have been recently investigated. These approaches maximize the likelihood of the data (corresponding to only maximizing H_g) directly by carefully designing a function that is both invertible and its Jacobian is easy to compute. These designs typically lead to approximations that either produce subpar samples or are inefficient. Alternatively, our approach does not require an invertible function. In addition, we apply the transformation in the dual space rather than directly on the input data. On the other hand, we know $\mathbb{E}_p \left[\mathbf{w}_1^\top \phi(\mathbf{w}_2, \mathbf{x}) \right] \leq A(\mathbf{w})$, then for a given distribution q we have, $\mathbb{E}_p \left[\phi(\mathbf{w}, \mathbf{x}) \right] - \mathbb{E}_{\mathbf{z} \sim p_z} \left[\phi(\mathbf{w}, g_{\boldsymbol{\theta}_g}(\mathbf{z})) \right] \leq H(p_z) + H_g$ where the entropies provide the bound on the difference between the moments under p and q.

3.3 Bayesian Extension

One benefit of such density estimation is that we can easily extend the model to a Bayesian one where a distribution over parameters given the observations is taken rather than the point estimate. As such, we employ prior distribution over the parameters and the Bayes rule to derive the posterior distribution over the parameters. For the exponential distribution in this paper we use a *conjugate* prior for parameter $\mathbf{w}_1$: $p(\mathbf{w}_1) = \exp(\mathbf{w}_1^\top \boldsymbol{\beta} - \beta_0 A(\mathbf{w}_1))$. The posterior is then computed in a closed-form: then

$$p(\mathbf{w}_1|\mathbf{x}) = \exp(\mathbf{w}_1^\top \boldsymbol{\beta} + \phi'(\mathbf{w}_2, \mathbf{x}) - (1 + \beta_0) A(\mathbf{w}))$$

For $\mathbf{w}_2$ we use the same prior, however, the posterior does not have a closed-form solution. As such, we employ Stochastic Gradient Langevin Dynamics (SGLD) (Welling and Teh 2011) to sample from this posterior. SGLD is an efficient algorithm that is easy to implement when the gradient of the log of the posterior is easy to compute (as it is in our case where the density is estimated using a deep neural network). SGLD performs Monte Carlo sampling by adding a Gaussian noise with a variance proportionate to the learning rate to the gradient updates of the log posterior. Therefore the predictive distribution for a test instance $\mathbf{x}^*$ is

$$p(\mathbf{x}^*|\mathcal{X}) = \int p(\mathbf{x}^*|\mathbf{w}) p(\mathbf{w}|\mathcal{X}) d\mathbf{w} \tag{9}$$

where we employ Monte Carlo estimate of this integral along with the samples of the posterior to estimate. Predicting the output in Eq. 9 needs to be weighted by the step size in the Langevin dynamics to adjust for the decrease in the step size which in turn decreases the mixing rate as explained in (Welling and Teh 2011).

Note in practice, we found it helps to speed up the "burn-in" process where we use Adam for the first few iterations (in larger datasets), after which we revert to SGD and sample according to SGLD update rule in Algorithm 1. As was shown in (Saatchi and Wilson 2017) such Bayesian approaches can significantly improve the performance due to the ability of the model in exploring the space and averaging parameter values.

3.4 Mode Collapse

Mode collapse is a phenomenon in which function $g_{\boldsymbol{\theta}_g}$ maps all $\mathbf{z}$s to a small number of points in the space of $\mathcal{X}$. This can happen in similar algorithms such as GANs where mode collapse is a major problem. Our approach is less likely to suffer from this phenomenon since we are maximizing the H_g for the generated samples. As shown in Fig. 1 we transform the input $\mathbf{z}$ to the output space as the local geometry of the

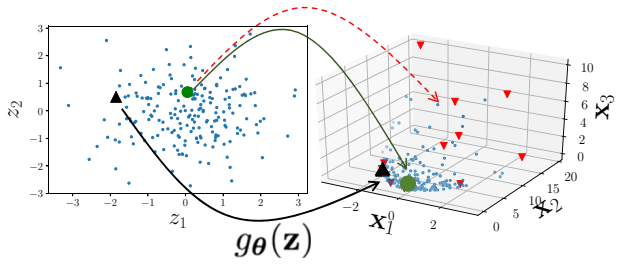


Fig. 1 Samples from $\mathbf{z}$ are transformed by $g(\mathbf{z})$. Red dashed-line indicate the transformation function that has higher $\log\det(J_\mathbf{z}^\top J_\mathbf{z})$. This transformation induces a higher entropy in the output space and hence better spread out

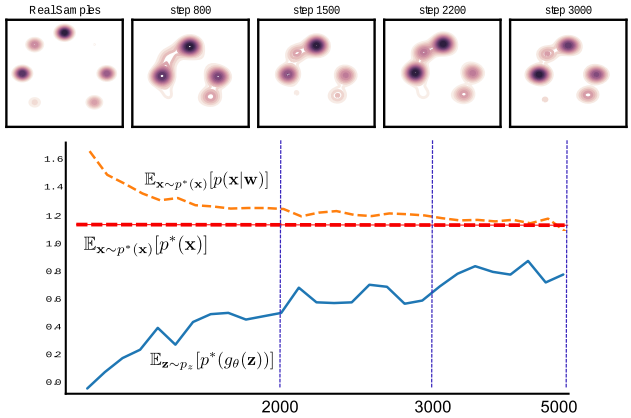


Fig. 2 A simple density estimation: the first row shows a multi-modal distribution and the distribution of samples from the generator at various stages; and, in the second diagram, the convergence of the estimated density to its true value is shown

output space better spread out because we are enforcing the local geometry of a point mapped in the output space $\mathbf{x}$ to encourage higher volume.

The interplay between the entropy of the distribution of $\mathbf{z}$ and the local characteristics of the transformation function heavily contributes to the performance of the quality of the samples generated. In particular, in case the distribution of $\mathbf{z}$ is Gaussian, maximizing the entropy amount to increasing its corresponding variance. Staring with a large variance and transforming to a spread-out space in p, leads to an estimate of the likelihood that covers all the space with relatively similar value. Hence, the samples generated this way exhibit poor quality and as such under-perform in estimating the correct likelihood. On the other hand, when the initial variance of the distribution of $\mathbf{z}$ is small and gradually this variance is increased, the samples generated by g_{θ_g} cover the mode of the distribution p first and subsequently learns the less frequent samples. In practice, we take the latter approach and start with a small variance for the p_z so that the mode of p is learnt first.

4 Experiments

Our approach provides a path for density estimation while we have the capacity to draw samples from the distribution learned from the dataset. As such, we direct our focus towards various applications where various aspects of our approach is evaluated. We use Algorithm 1 to train our density model in which a generator is also learnt that we can sample from. When the distribution of the output for the generator is too large, computing the Jacobian is expensive. As such, during training we sample a set of dimensions from the output and compute the gradients. To estimate the Jacobian we use Lemma 2 and found the value of α set to the dimension of the Jacobian performs well in practice. In the interest of efficiency, we used $k = 2$. We use the Jacobian-vector trick similar to that of (Martens 2010) to compute the Jacobian for which we used 5 samples. Note that these 5 samples, are just matrix-vector products and as such very efficient. In addition, we update the parameter $\mathbf{w}$ twice as often as that of $\boldsymbol{\theta}$ while regularize the gradients to ensure stable training that converges to an optimum solution.

4.1 Simulation

For the first experiment, we use samples from a Gaussian mixture model to simulate a dataset we are interested in its density model. Each component is a Gaussian distribution with standard deviation 0.1. The generated dataset is shown in Fig. 2 on top. We then use a simple two fully connected layer neural network for the generator and $\phi(\mathbf{w}, \mathbf{x})$. In the top row of Fig. 2 the generated samples are plotted and as shown over time our approach is able to capture the density model and generate samples from all the modes. Furthermore, at the bottom of Fig. 2 the convergence of the density estimate to its true value is shown. As observed, as the distribution of the samples from the generator look more similar to the real ones, the estimate of the density (as measured by the expected log-likelihood with respect to the true distribution) approaches the true value.

4.2 Image Generation and Density Estimation

In this section, we examine the quality of the generated images using our approach as well as its density estimation ability. We use DCGAN's (Radford et al. 2015) architecture for image generation.

MNIST To evaluate the effect of the Jacobian on the mode collapse, we have conducted an experiment similar to that of (Metz et al. 2016) where we select 3 digits at random and build a 3-channel "stacked-MNIST" by concatenating them. Using a pre-trained classifier, we find the number of modes for which the generator produced at least one sample. The

Table 1 Mode collapse on Stacked-MNIST

	Modes
DCGAN (Radford et al. 2015)	99
ALI (Dumoulin et al. 2017)	16
Unrolled GAN (Metz et al. 2016)	48.7
GADE	125
GADE Bayesian	**142**

Table 2 MNIST test set negative log-likelihood with generative models (lower is better). These values are estimates since the true density is intractable

MNIST	$-\log p(\mathbf{x})$
VAE (Kingma and Welling 2014)	87.88
IWAE ($IW = 50$) (Burda et al. 2015)	86.10
VAE+NF (Rezende and Mohamed 2015))	85.10
ASY-IWAE (Zheng et al. 2017)	**85.76**
Auxiliary VAE (Maaløe et al. 2016)	84.59
DARN (Gregor et al. 2014)	84.13
IWAE+ConvFlow (Zheng et al. 2018)	79.78
GADE	82.39
GADE Bayesian	83.12

Fig. 3 Samples from MNIST sorted based on their density from high to low

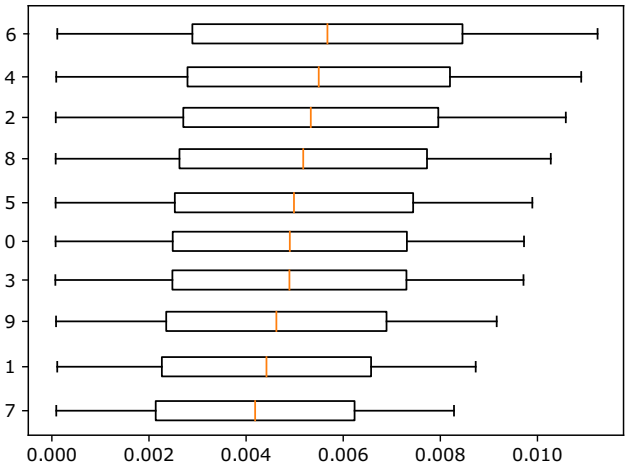
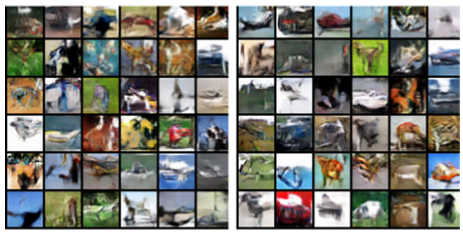


Fig. 4 Average posterior probability for each digit sorted from the highest variance to the lowest

(a) Max LL **(b)** Bayesian



(c) Max LL **(d)** Bayesian

Fig. 5 Samples drawn from the generator using MNIST: **a** and **b**; and, CIFAR-10 **c** and **d**. Alternatives (e.g. (Kolesnikov and Lampert 2017; Salimans et al. 2017)) can generate visually better quality images

results of running this experiment is shown in Table 1. As observed our approach outperforms its counterparts indicating the ability of our approach in avoiding mode-collapse.

We use MNIST, a dataset of handwritten characters, to generate samples from q. We show samples both from maximum log-likelihood and Bayesian. We use the average predictive probability for each instance (obtained similar to that of Eq. 9) and show the results in Fig. 5. It is interesting to observe that 1 and 7 are the least likely digits because they have straight lines that have structural differences with other digits. In addition, we compare the likelihood estimated by our approach compared to the-state-of-the-art. The evaluation of log-likelihood (LL) in nats is reported in Table 2. All log-likelihood values were estimated as the mean of 5000 instances drawn randomly from the test set. Since we use the predictive distribution in Eq. 9, the likelihood is not as low as when we compute the likelihood directly. Hence, the reported values are stochastic lower bounds on the true value.

Figure 3, 4 depicts samples drawn from the generator sorted by the likelihood using our Bayesian approach.

For MNIST, we have the variance of $\phi(\mathbf{w}, g_{\theta_g}(\mathbf{z}))$ for 500 instances when using WGAN-GP to be 0.009 while ours is 0.03 (which is 3.3 times more). Besides, for the complete dataset when we use both real and generated samples, this variance changes from 0.3 in WGAN-GP to 5.28 in ours. That is an indication that our approach–rather than simply distinguishing real from fake–learns the likelihoods.

CIFAR-10 We have trained our approach on CIFAR-10 dataset (Krizhevsky and Hinton 2009) composed of 50, 000 RGB images of natural scenes. Samples from this dataset is shown in Fig. 5. We achieve the inception score of 7.84 using a modified ResNet.

CelebA While MNIST and CIFAR-10 are smaller dimensions, we evalue our approach on CelebA (Liu et al. 2015) with size 256 × 256. While our approach performs density estimation, we can generate high-quality images to estimate

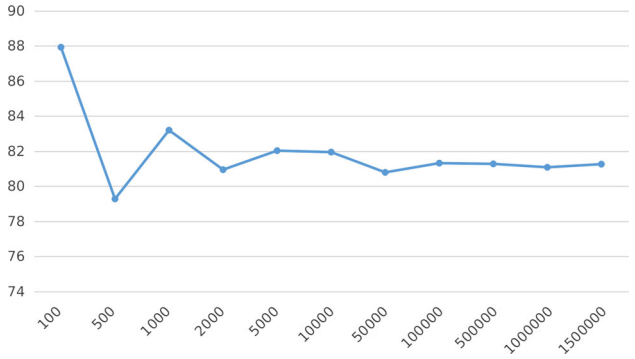


Fig. 6 Estimation of the normalizer based on the number of samples. Increasing the number of samples improves the estimation and approaches the true expectation for the normalizer of MNIST

the normalizer. In this experiment, we utilize the Progressive-GAN (Karras et al. 2017) architecture and learning procedure to produce images and evaluate their density estimation. We compare the output of our density estimator versus WGAN-GP's (Gulrajani et al. 2017) discriminator output as shown in Figs. 6, 7. The histogram distribution of the fake samples in WGAN-GP looks very similar to the Normal distribution similar to the prior noise. In addition, as expected with most GAN-based methods, the discriminator value overlaps between real and fake samples indicating the discriminator's confusion. Our approach on the other hand, shows a better spread histograms. Furthermore, the images in the graph shows samples of generated images with their probability as computed by our approach. It is observed that more typical looks that are more representative of the dataset concentrated in a high density area. In the progressive training, we freeze the update of the noise entropy for the higher quality images to avoid changing the networks significantly. The Frechet Inception Distance (FID) computed from 50K images is 5.96 for this dataset.

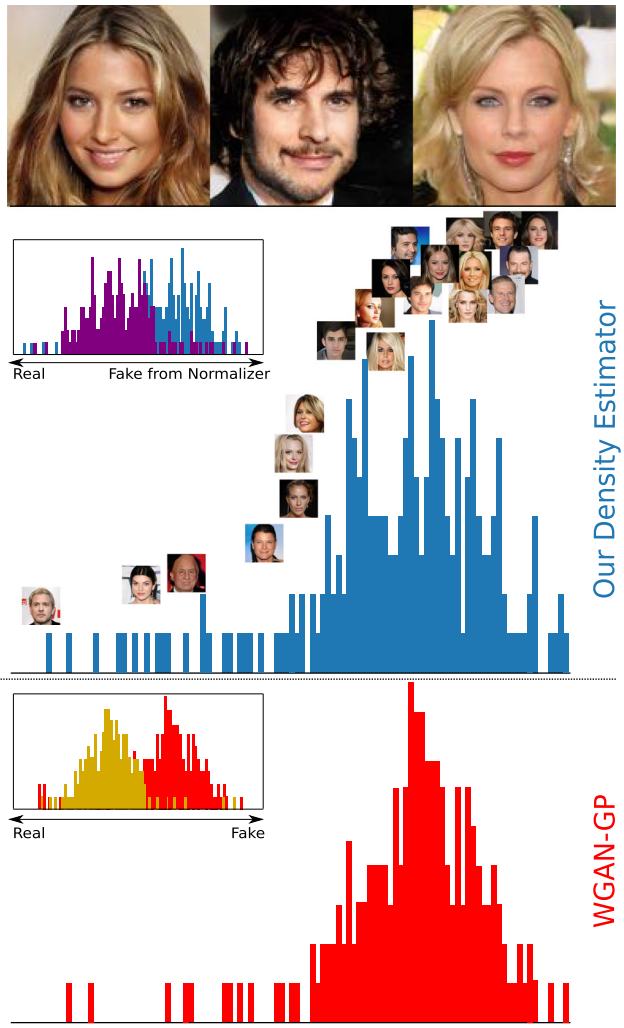


Fig. 7 Histogram of the evaluation of $\phi(\mathbf{w}, g_{\theta_g}(\mathbf{z}))$ for 500 samples generated from the CelebA dataset with 256 × 256 dimensions (on top) compared to the discriminator value of WGAN-GP (on bottom). On the left corner of each plot, the distribution of the $\phi(\mathbf{w}, g_{\theta_g}(\mathbf{z}))$ is shown vs. the output of the discriminator in the WGAN-GP. As shown, discriminator in WGAN-GP finds a distribution that has around half overlap between reals and the fakes

4.3 Uncertainty in Visual Question Answering

We utilize our approach in the visual dialog problem introduced in (Das et al. 2017). The task is to design an agent capable of replying to questions based on image content and history of a dialog with a human user. The objective is to generate responses to questions that are most likely to be given by a human. The transformation function g_{θ_g} is responsible for generating a suitable answer for the user's question, given the previous history of agent-user conversations, the user question and the visual content (rather than just a noise distribution in the previous experiment). Then, the likelihood of the human responses is higher.

Our approach is well embedded into a larger dialog system built upon that of (Serban et al. 2016; Lu et al. 2017). We use a hierarchical setup where a neural network for sequence models (LSTM in our case) encodes the word sequence and another one captures the context through historical dependencies of the conversions. A common practice is to use an attention mechanism with the encoder to implicitly identifying parts of dialog history relevant to the current question (Serban et al. 2016; Das et al. 2017). It is expected that a similar mechanism is able to localize the relevant regions in the image consistent with the attended history. Since the output is discrete, we use Gumbel-Max trick (Jang et al. 2016; Maddison et al. 2016) which illustrates the ability of our approach in maximizing likelihood for discrete data.

This task is specifically suitable for Bayesian formulation of our approach since the space in which the function g_{θ_g} formulates the responses is large and as such this causes

Table 3 Results on visual dialog dataset with late fusion (LF), hierarchical recurrent encoder (HRE), memory network (MN). GAN1 is devised by using generated fake samples in the discriminator and GAN2, in addition, includes negative answers

	Discriminative					Generative				
Model	MRR	R@1	R@5	R@10	Mean	MRR	R@1	R@5	R@10	Mean
HieCoAtt (Lu et al. 2016)	0.5788	43.51	74.49	83.96	5.84	–	–	–	–	–
LF (Das et al. 2017)	0.5807	43.82	74.68	84.07	5.78	0.5199	41.83	61.78	67.59	17.07
HRE (Das et al. 2017)	0.5868	44.82	74.81	84.36	5.66	0.5242	42.28	62.33	68.17	16.79
MN (Das et al. 2017)	0.5965	45.55	76.22	85.37	5.46	0.5259	42.29	62.85	68.88	17.06
HCIAE (Lu et al. 2017)	0.6140	**47.73**	77.50	86.35	5.15	0.5386	44.06	63.55	69.24	16.01
GAN1 (Lu et al. 2017)	0.2177	8.82	32.97	52.14	18.53	0.5298	43.12	62.74	68.58	16.25
GAN2 (Lu et al. 2017)	0.6050	46.20	77.92	87.20	4.97	0.5459	44.33	65.05	71.40	14.34
WGAN	0.5524	43.77	67.75	72.32	13.18	0.5306	43.49	62.38	67.73	17.15
Ours	0.6125	47.17	77.25	87.65	4.82	0.5448	44.37	64.34	71.81	14.93
Ours Bayesian	**0.6204**	47.65	**78.91**	**88.30**	**4.61**	**0.5513**	**45.03**	**65.25**	**71.93**	**14.01**

Higher values are better except for the mean. Best numbers are boldfaced

uncertainty in the prediction. Hence, our approach leverages this uncertainty in the predictive distribution when responding by providing "I don't know" statements when the predictive variance is high.

The visual dialog (Das et al. 2017) dataset we use was collected by pairing two Amazon Mechanical Turk users conversing about an image. The "questioner" user is shown the captions of the image (from COCO dataset (Lin et al. 2014)) while the image itself remains hidden. The "answerer" user sees the image and provides responses about the image to the questioner. This dialog is carried out for 10 rounds on the COCO dataset leading to 83, 000 training dialogs and 40, 000 validation dialog. Following the baseline, the likelihood score is used in ranking amongst 100 candidate answers.

We set the parameter for the Gumbel temperature to 0.1. Our LSTMs are single layer with 512-dimensional hidden states. Further, we use VGG-19 (Simonyan and Zisserman 2014) to represent images and Adam optimization with the learning-rate of 4e-4. For the Bayesian case, we use SGD and set the learning rate to decrease from 1e-3 to 1e-4. Following the problem setup in (Lu et al. 2017), we pre-train our transformation function g_{θ_g} by maximizing the score it generates compared to the ground-truth. Similarly, we pre-train the density estimator using samples from 100 rounds of the question-answers such that the score of the ground-truth is higher than non-matching responses in the training set. We pre-train both for 20 epochs. Moreover, we regularize the latent space from which the answers are generated with its entropy (we assumed the latent space is a zero-mean Gaussian). We draw 5 samples from which we pick the most likely.

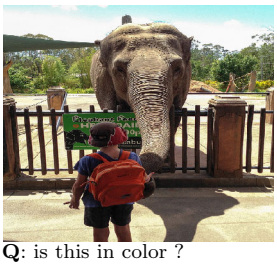
In Table 3 we provide the results of running our approach on the visual dialog dataset compared to the state-of-the-art GAN methods and other baselines in (Shetty et al. 2017; Lu et al. 2017). Models are evaluated on standard retrieval metrics: mean rank, recall @k, and mean reciprocal rank (MRR) of human response. The results from the pre-training stage is indicated as HCIAE in the table and as is shown GAN training improves the performance. In addition, our approach outperforms the baseline with a good margin and is more stable in training due to its regularization in the output. Moreover, we devised the Bayesian variant (as in Sect. 3.3) to compute the predictive distribution and pick the one with the highest confidence according to Eq. 9 (highest mean plus standard deviation). Since the space of answers generated is typically large even with pre-training, performance of the generator and discriminator can deteriorate. This is because the noisy samples from the generator lead to decreased performance in the discriminator.

In Table 4 we show samples from the visual dialog dataset. We evaluate the generated answers by sampling the discriminator function and evaluating the variance of the prediction (likelihood of the generated answer being human-response in the log space). As is shown in the table, we observe the variance of the prediction is generally higher for the wrong responses. We can use a simple thresholding on the predictive variance to determine the answers to be substituted by "I don't know". For instance, in Table 4 we show an example where a highly uncertain answer is substituted. Even though for frequent answers such as yes/no responses the variances are less reliable. In addition, for longer sentences, the variances are generally higher. This is expected due to the potential diversity.

4.4 Imitation Learning for Autonomous Driving

Success in reinforcement learning heavily relies on the effectiveness of the reward function Ramachandran and Amir (2007); Levine and Koltun (2013). Imitation learning meth-

Table 4 Qualitative comparison from the visual dialog dataset

		
Q: is the train old? **A**: i think so **Sample A**: ~~yes~~ (0.03 ± 0.2) **Q**: is the train moving? **A**: no but it does have some steam coming **Sample A**: no (0.02 ± 0.1)	**Q**: is this in color ? **A**: yes **Sample A**: yes (0.1 ± 0.05) **Q**: how old does the boy look? **A**: he is not facing me, but maybe **Sample A**: ~~3~~ (0.1 ± 0.4)	**Q**: what vegetables are there? **A**: carrots, cauliflower, broccoli **Sample A**: carrots , carrots , and cucumbers (0.1 ± 0.01) **Q**: what color is the table ? **A**: dark brown **Sample A**: white (0.1 ± 0.02)

(Q) is the question, (A) is the ground-truth human response, (Sample A): is the generated answer from the generator. Red response indicates lack of confidence, which will be substituted by "I don't know"

ods are designed to learn this reward function by utilizing expert demonstrations and has succeeded in a wide range of problems Ziebart et al. (2008); Englert and Toussaint (2015); Stadie et al. (2017). Behavioral cloning that casts this problem as a supervised learning task typically demonstrates poor generalization performance and as such imitation learning has gained momentum.

Recently proposed Generative Adversarial Imitation Learning (GAIL, (Ho and Ermon 2016)) is a model-free imitation learning method that is highly effective and scales to relatively high dimensional environments. GAIL builds upon a generative model: the stochastic policy coupled with a fixed simulation environment produces behaviors as similar as possible to that of the expert demonstrations. The discriminator distinguishes the expert trajectories from ones produced by the generator's as in GANs.

Human is typically the expert that produces the examples for the imitation learning. The high-dimensional nature of these examples along with the variability in expert's demonstrations make this problem challenging. Specifically when the input is visual, the policy needs to simultaneously learn how to identify meaningful visual features, and how to leverage them for achieving desired behaviors using a small set of expert demonstrations. In addition, evaluation of the likelihood of the policies allows better reward formulation which is challenging in such problems. In addition, since such learnt rewards are uncertain in nature being able to quantify the confidence in predictions allows for actions that are more desirable.

It should be noted that the generator's objective in our approach can produce particularly ineffective trajectories because the generator seeks to model low-frequency areas as well. This means when the policy makes a small mistake, the generator deviates from the state distribution seen during training, making it more likely to make a mistake again. In particular, when the expert data has a larger variance. One way for handling this problem is to gradually sample more from the generator rather than data during training. In addition, we draw 5 samples from the generator and pick the most likely one.

For experiments, similar to Li et al. (2017), we consider an autonomous driving example implemented using TORCS driving simulator Espié et al. (2000). We evaluate two subsets of human driving behaviors: *turn* or *pass*. Turn behavior can be either inside or outside the land and pass is either from the left or right. In both cases, the expert's policy has two significant modes. Our objective is for the algorithm to be able to distinguish the modes in the expert's demonstrations.

We use a pre-trained deep residual net (He et al. 2016) to extract the visual features and use as inputs for the policy network. Similar to Li et al. (2017), we use a discrete value to model turn or pass policies and optimize its mutual information based on InfoGAN's method (Chen et al. 2016). For both settings, there are 80 expert trajectories in total, with 100 frames in each trajectory. In the initial stage, both passes and turns are equally likely (uniform distribution). The generator produces samples of the state-action pairs for the density estimator to evaluate their likelihood. Subsequently, the likelihood of the trajectories being expert's is maximized through which we estimate the normalizer and learn to produce high-quality trajectories in the generator. The samples in this experiment are the state-action pairs. Our approach uses trust region policy optimization (TRPO) Schulman et al. (2015) in updating the generator parameters that allows for better policies.

The performance of policies is evaluated using two metrics: *accuracy* as the percentage of correctly predicted expert's state-action pairs, and the *average distance* which

Table 5 Classification accuracy for pass trajectories

Method	Accuracy
PCA Li et al. (2017)	61.7%
InfoGAIL Li et al. (2017)	81.9%
GADE	81.6%
GADE Bayesian	**82.6**%

They are the results from our method and have the highest value

Table 6 Average rollout distances for imitation learning

Method	Avg. rollout distance
Behavior cloning	701.83
GAIL (Ho and Ermon 2016)	914.45
WGAN	1177.72
InfoGAIL (Li et al. 2017)	1226.68
GADE	1223.13
GADE Bayesian	**1228.21**
Human	1203.51

They are the results from our method and have the highest value

determines the distance traveled by the agent before a collision. We perform reward augmentation similar to (Li et al. 2017) to discourage the car from crashing early.

We sample $\mathbf{z}_i$s with state-action pairs from the expert trajectories and calculate the accuracy of the correctly predicted state. The results are shown in Table 5. Interestingly, without supervised training, our approach correctly classifies expert trajectories. The average rollout distances are shown in Table 6. Our method is able to outperform the expert while WGANs perform worse.

5 Conclusion

In this paper, we introduced generative adversarial density estimator. Our approach estimates the density of data using a lower bound on its normalizer. We formalized the problem as a maximum likelihood for the density estimator in which a transformation function generates samples for approximating the normalizer. We showed that having a density model and the prior on the parameters, we are able to build a Bayesian alternative using which we can measure our confidence in the likelihood. Our experiments on challenging applications such as image density or visual dialog where both the likelihood and confidence in predictions are crucial shows the effectiveness of our approach. Moreover, the samples drawn from the estimated density is less susceptible to mode collapse which is a desired property for generative models.

We believe our approach opens new avenues for further research in density estimation and its marriage with adversarial models.

Proof of Lemma 2

Proof Since B is a positive definite matrix with $\|B\|_2 < 1$, it follows that,

$$\begin{aligned}\log\det A &= \log(\alpha^d \det(A/\alpha)) \\ &= d\log(\alpha) + \log\left(\prod_{i=1}^{d}\lambda_i(B)\right) \\ &= d\log(\alpha) + \sum_{i=1}^{d}\log(\lambda_i(B)) = d\log(\alpha) + \mathrm{tr}\,(\log(B))\,.\end{aligned}$$

where $\lambda_i(B)$ is the ith eigenvalue of the matrix B. We know for the second line that

$$\mathrm{tr}\,(\log(B)) = \sum_{i=1}^{d}\lambda_i(\log(B)) = \sum_{i=1}^{d}\log(\lambda_i(B)).$$

Then we have,

$$\mathrm{tr}\,(\log(B)) = \mathrm{tr}\,(\log((B-\mathbf{I}_d)+\mathbf{I}_d)) \tag{10}$$

$$= \mathrm{tr}\left(\sum_{k=1}^{\infty}\frac{(-1)^{k+1}(B-\mathbf{I}_d)^k}{k}\right). \tag{11}$$

For the second equality we use Taylor expansion of the log since all the eigenvalues of $B-\mathbf{I}_d$ are in $(0, 1)$ and the last equality follows by the linearity of the trace operator. □

References

Arjovsky, M., Chintala, S., Bottou, L. (2017). Wasserstein gan. arXiv:1701.07875.

Ben-Israel, A. (1999). The change-of-variables formula using matrix volume. *SIAM Journal on Matrix Analysis and Applications*, *21*(1), 300–312. https://doi.org/10.1137/S0895479895296896.

Boutsidis, C., Drineas, P., Kambadur, P., Kontopoulou, E. M., & Zouzias, A. (2017). A randomized algorithm for approximating the log determinant of a symmetric positive definite matrix. *Linear Algebra and its Applications*, *533*, 95–117.

Burda, Y., Grosse, R., Salakhutdinov, R. (2015). Importance weighted autoencoders

Chen X, Duan Y, Houthooft R, Schulman J, Sutskever I, Abbeel P (2016) Infogan: Interpretable representation learning by information maximizing generative adversarial nets. CoRR abs/1606.03657

Das, A., Kottur, S., Gupta, K., Singh, A., Yadav, D., Moura, J.M., Parikh, D., & Batra, D. (2017). Visual Dialog. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Diggle, P. J., & Gratton, R. J. (1984). Monte Carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society*, *46*, 193–227.

Dinh, L., Krueger, D., & Bengio, Y. (2014). Nice: Non-linear independent components estimation. *CoRR*,. abs/1410.8516.

Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2016). Density estimation using real nvp.

Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., & Courville, A. (2017). Adversarially learned inference. In *International conference on learning representation*.

Englert, P., & Toussaint, M. (2015). Inverse kkt–learning cost functions of manipulation tasks from demonstrations. In *Proceedings of the international symposium of robotics research*.

Espié, E., Wymann, B., Dimitrakakis, C., Guionneau, C., Coulom, R., & Sumner, A. (2000). Torcs, the open racing car simulator. http://torcs.sourceforge.net.

Finn, C., Christiano, P.F., Abbeel, P.,& Levine, S. (2016). A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. CoRR, arXiv:1611.03852

Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.,& Bengio, Y. (2014). Generative adversarial nets. In *International conference on neural information processing systems*.

Gregor, K., Danihelka, I., Mnih, A., Blundell, C., & Wierstra, D. (2014). Deep autoregressive networks. In *The international conference on machine learning (icml)*.

Grover, A., Dhar, M., & Ermon, S. (2018). Flow-gan: Combining maximum likelihood and adversarial learning in generative models. In *Proceedings of the thirty-second AAAI conference on artificial intelligence, (AAAI-18), the 30th innovative applications of artificial intelligence (IAAI-18), and the 8th AAAI symposium on educational advances in artificial intelligence (EAAI-18), New Orleans, Louisiana, USA*, February 2–7, 2018 (pp. 3069–3076).

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. *CoRR*,. abs/1704.00028.

Gutmann, M. U., Dutta, R., Kaski, S., & Corander, J. (2018). Likelihood-free inference via classification. *Statistics and Computing*, *28*, 411–425.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Ho, J.,& Ermon, S. (2016). Generative adversarial imitation learning. In *Advances in neural information processing systems*.

Jang, E., Gu, S., & Poole, B. (2016). Categorical reparameterization with gumbel-softmax. CoRR.

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. In *The international conference on learning representations (ICLR)*.

Kingma, D.P., Welling, M. (2014). Auto-encoding variational bayes. In *The international conference on learning representations (ICLR)*.

Kolesnikov, A.,& Lampert, C. H. (2017). PixelCNN models with auxiliary variables for natural image modeling. In Precup D., & Teh Y.W. (eds) *Proceedings of the 34th international conference on machine learning, PMLR, international convention centre, Sydney, Australia, Proceedings of Machine Learning Research* (Vol. 70, pp. 1905–1914).

Konda, V. R., & Tsitsiklis, J. N. (2004). Convergence rate of linear two-time-scale stochastic approximation. *The Annals of Applied Probability*, *14*(2), 796–819.

Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

Levine, S., & Koltun, V. (2013). Guided policy search. In *International conference on machine learning*.

Li, Y., Song, J., & Ermon, S. (2017). Infogail: interpretable imitation learning from visual demonstrations. arXiv preprint arXiv:1703.08840.

Lin, T. Y., Michael, M., Serge, B., James, H., Pietro, P., Deva, R., Piotr, D., & Lawrence, Z. C. (2014). Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014*, Berlin: Springer International Publishing.

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of international conference on computer vision (ICCV)*.

Lu, J., Yang, J., Batra, D., & Parikh, D. (2016). Hierarchical question-image co-attention for visual question answering. In *Proceedings of the 30th international conference on neural information processing systems, Curran Associates Inc., USA, NIPS'16* (pp 289–297). http://dl.acm.org/citation.cfm?id=3157096.3157129.

Lu, J., Kannan, A., Yang, J., Parikh, D., Batra, D. (2017). Best of both worlds: Transferring knowledge from discriminative learning to a generative visual dialog model. NIPS.

Maaløe, L., Sønderby, C. K., Sønderby, S. K., Winther, O. (2016). Auxiliary deep generative models. In *Proceedings of the 33rd international conference on international conference on machine learning, ICML'16* (pp. 1445–1454).

Maddison, C. J., Mnih, A., Teh, Y. W. (2016). The concrete distribution: A continuous relaxation of discrete random variables. CoRR.

Martens, J. (2010). Deep learning via hessian-free optimization. In *Proceedings of the 27th international conference on international conference on machine learning*, Omnipress, Madison, WI, USA, ICML–10, pp. 735–742.

Metz, L., Poole, B., Pfau, D., & Sohl-Dickstein, J. (2016). Unrolled generative adversarial networks. *International Conference on Learning*,. Representations.

Nguyen, A., Clune, J., Bengio, Y., Dosovitskiy, A., & Yosinski, J. (2017). Plug & play generative networks: Conditional iterative generation of images in latent space. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, IEEE.

Nowozin, S., Cseke, B., & Tomioka, R. (2016). f-GAN: Training generative neural samplers using variational divergence minimization. arXiv:1606.00709.

Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., Kavukcuoglu, K. (2016). Conditional image generation with pixelcnn decoders. In *Proceedings of the 30th international conference on neural information processing systems, NIPS'16* (pp. 4797–4805).

Pfau, D., & Vinyals, O. (2016). Connecting generative adversarial networks and actor-critic methods. CoRR. abs/1610.01945.

Price, B., & Boutilier, C. (2003). A bayesian approach to imitation in reinforcement learning. In *Proceedings of the 18th international joint conference on artificial intelligence* IJCAI'03 (pp. 712–717).

Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. CoRR

Ramachandran, D., Amir, E. (2007). Bayesian inverse reinforcement learning. In *Proceedings of the 20th international joint conference on artifical intelligence, IJCAI'07* (pp. 2586–2591).

Rezende, D.J., & Mohamed, S. (2015). Variational inference with normalizing flows. In *Proceedings of the 32nd international conference on international conference on machine learning*, JMLR.org, ICML'15 (pp 1530–1538).

Saatchi, Y., & Wilson, A. G. (2017). Bayesian GAN. In *Advances in neural information processing systems*.

Salakhutdinov, R., & Hinton, G.E. (2009). Deep boltzmann machines. In *International conference on artificial intelligence and statistics,* AISTATS'09.

Salimans, T., Karpathy, A., Chen, X., & Kingma, D.P. (2017). Pixelcnn++: A pixelcnn implementation with discretized logistic

mixture likelihood and other modifications. In *The international conference on learning representations (ICLR)*.

Schulman, J., Levine, S., Moritz, P., Jordan, M.,& Abbeel, P. (2015). Trust region policy optimization. In *Proceedings of the 32nd international conference on international conference on machine learning* (Vol. 37, pp 1889–1897).

Serban, I.V., Sordoni, A., Bengio, Y., Courville, A.,& Pineau, J. (2016). Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings AAAI* (pp .3776–3783).

Shetty, R., Rohrbach, M., Hendricks, L. A., Fritz, M., & Schiele, B. (2017). Speaking the same language: Matching machine to human captions by adversarial training. *CoRR*,. abs/1703.10476.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv.

Stadie, B., Abbeel, P., & Sutskever, I. (2017). Third person imitation learning. In *International conference in learning representation*.

Theis, L., van den Oord, A., & Bethge, M. (2015). A note on the evaluation of generative models. *ArXiv e-prints*, *1511*, 01844.

Wang, T. C., Liu, M. Y., Zhu, J. Y., Tao, A., Kautz, J., & Catanzaro, B. (2017). High-resolution image synthesis and semantic manipulation with conditional gans.

Welling, M., & Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on international conference on machine learning* (pp 681–688).

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., & Metaxas, D. (2017). Stackgan: text to photo-realistic image synthesis with stacked generative adversarial networks. In: ICCV.

Zheng, G., Yang, Y., & Carbonell, J. (2017). Likelihood almost free inference networks.

Zheng, G., Yang, Y., & Carbonell, J. (2018). Convolutional normalizing flows. In *ICML workshop on theoretical foundations and applications of deep learning*.

Ziebart, B. D., Maas, A., Bagnell, J. A.,& Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd national conference on artificial intelligence* (Vol. 3, AAAI'08).