Variational Generative Stochastic Networks with Collaborative Shaping

Philip Bachman PHIL.BACHMAN@GMAIL.COM
McGill University, School of Computer Science

Doina Precup DPRECUP@CS.MCGILL.CA
McGill University, School of Computer Science

Abstract

We develop an approach to training generative models based on unrolling a variational auto-encoder into a Markov chain, and shaping the chain's trajectories using a technique inspired by recent work in Approximate Bayesian computation. We show that the global minimizer of the resulting objective is achieved when the generative model reproduces the target distribution. To allow finer control over the behavior of the models, we add a regularization term inspired by techniques used for regularizing certain types of policy search in reinforcement learning. We present empirical results on the MNIST and TFD datasets which show that our approach offers state-of-the-art performance, both quantitatively and from a qualitative point of view.

1. Introduction

Significant research effort has been directed towards developing models capable of effectively synthesizing samples from complicated distributions. We propose an approach to this problem whose goal is two-fold. We want to learn a distribution which is practically indistinguishable from the target distribution and we also want training, inference and sampling to be efficient. Our approach can be viewed as a class of Generative Stochastic Networks (GSNs) (Bengio et al., 2014). We show that any model trained with the *walkback* procedure (Bengio et al., 2013) is encompassed by our approach. Instead of using denoising auto-encoders, we leverage recent variational methods for deep, directed generative models, e.g. (Kingma & Welling, 2014; Rezende et al., 2014; Mnih & Gregor, 2014), and build our approach starting from variational auto-encoders. By feeding the output of such an auto-encoder back into itself, we construct a Markov chain whose stationary distribution provably (in the non-parametric, infinite-data limit) converges to the target distribution. A bound on the log-density of the model distribution is also efficiently computable.

As an alternative to the walkback procedure for training GSNs, we propose an approach based on recent work in Approximate Bayesian Computation. We partner a generative model with a function approximator that estimates the log-density ratio between the model-generated distribution and the target distribution, in what can be seen as a *collaborative* alternative to the *adversarial* approaches in (Gutmann et al., 2014a;b; Goodfellow et al., 2014). We show that the global minimizer of the resulting objective (in the non-parametric, infinite-data limit) is achievable only when the model distribution matches the target distribution.

To control the model complexity, we introduce a regularization term close in spirit to reinforcement learning methods such as relative entropy policy search (Peters et al., 2010) and other approaches which depend on a notion of "natural system dynamics", e.g. (Todorov, 2009). Specifically, we re-weigh the standard $\mathrm{KL}(\text{posterior} \,||\, \text{prior})$ term that appears in the variational free-energy. Because in this type of model, almost all complexity in $p(x) = \sum_z p(x|z)p(z)$ is captured by the value of the latent variables $z \in \mathcal{Z}$, this approach provides a direct mechanism to trade off the complexity of $p(x)$ against the ability to exactly reproduce the training distribution.

Our models permit efficient generation of independent samples, efficient generation of sequences of samples representing "locally-coherent" random walks along the data manifold, and efficient evaluation of a variational lower-bound on the log-likelihood assigned to arbitrary inputs. We show that our approach produces models which significantly outperform the GSNs in (Bengio et al., 2014) and the adversarial networks in (Goodfellow et al., 2014) in terms of test-set log-likelihood and qualitative behavior.

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

2. Background

This section provides a summary of prior work on denoising auto-encoders and Generative Stochastic Networks, which constitute the basis of our model.

2.1. Generalized Denoising Auto-encoders

In the Generalized Denoising Auto-encoder (DAE) framework (Bengio et al., 2013), one trains a *reconstruction distribution* $p_\theta(x|\tilde{x})$ to match the conditional distribution $\mathcal{P}(x|\tilde{x})$ implicit in an infinite set of pairs $\{(x_1, \tilde{x}_1), ..., (x_n, \tilde{x}_n)\}$ generated by drawing each $x_i \in \mathcal{X}$ from the *target distribution* $\mathcal{D}$ and then generating each $\tilde{x}_i \in \mathcal{X}$ by applying some stochastic *corruption process* $q_\phi(\tilde{x}|x)$ to x_i. Given q_ϕ and p_θ, where θ and ϕ denote parameters of the two distributions, one can construct a Markov chain over $x \in \mathcal{X}$ by iteratively sampling a point x_t from $p_\theta(x_t|\tilde{x}_{t-1})$ and then sampling a point $\tilde{x}_t$ from $q_\phi(\tilde{x}_t|x_t)$. The chain is initialized at $t = 0$ by sampling x_0 directly from $\mathcal{D}$ and its transition operator $T_\theta(x_t|x_{t-1})$ can be computed by marginalizing over $\tilde{x}_{t-1}$.

Given a few small assumptions on the forms of q_ϕ and p_θ, and the larger assumption that $p_\theta(x|\tilde{x})$ provides a consistent estimator of $\mathcal{P}(x|\tilde{x})$ as the number of training samples $x \sim \mathcal{D}$ goes to infinity, it was shown in (Bengio et al., 2013) that the Markov chain constructed from the iterative process described above will be ergodic and have a stationary distribution π_θ which matches $\mathcal{D}$ (i.e. $\forall x, \pi_\theta(x) = \mathcal{D}(x)$).

All of the discussion in (Bengio et al., 2013) assumed that both x_i and $\tilde{x}_i$ for each training pair $(x_i, \tilde{x}_i)$ were from the same space $\mathcal{X}$, although this was not required for their proofs. The Generative Stochastic Network (GSN) framework (Bengio et al., 2014) thus made the jump of assuming a corruption process $q_\phi(z_t|x_{t-1}, z_{t-1})$. This extends the Generalized DAE framework by introducing a *latent* space $\mathcal{Z} \neq \mathcal{X}$, and by allowing the current latent state z_t to depend on the previous latent state z_{t-1} (in addition to its dependence on the previous *observable* state x_{t-1}). Figures 2 (a) and (b) illustrate the graphical models corresponding to Generalized DAEs and GSNs.

2.2. Training with Walkback

A method called *walkback* training was proposed for Generalized DAEs in (Bengio et al., 2013) and used again for GSNs in (Bengio et al., 2014). The motivation for walkback training was to mitigate difficulties encountered in practical, finite-data settings, where many values for the latent variables $z \in \mathcal{Z}$ that were rarely (if ever) visited during training would appear when sampling from the resulting Markov chain. The difficulties stem primarily from a desire to make the corruption process q_ϕ induce a conditional distribution $\mathcal{P}(x|z)$ which is roughly unimodal over x given any particular z (because this makes it easier to model with $p_\theta(x|z)$), which fights against the possibility that $\mathcal{D}$ contains multiple well-separated modes (which would necessitate a relatively non-local corruption process, able to "carve out" reliable paths between these modes during training).

Algorithm 1 Walkback for a General GSN

Input: data sample x, corruptor q_ϕ, reconstructor p_θ
Initialize an empty training pair list $\mathcal{P}_{xz} = \{\}$
Set $\hat{z}$ to some initial vector in $\mathcal{Z}$.
for $i = 1$ **to** $k_{burn-in}$ **do**
  Sample $\check{z}$ from $q_\phi(z|x, \hat{z})$ then set $\hat{z}$ to $\check{z}$.
end for
Set $\hat{x}$ to x.
for $i = 1$ **to** $k_{roll-out}$ **do**
  Sample $\check{z}$ from $q_\phi(z|\hat{x}, \hat{z})$ then set $\hat{z}$ to $\check{z}$.
  Sample $\check{x}$ from $p_\theta(x|\hat{z})$ then set $\hat{x}$ to $\check{x}$.
  Add pair $(x, \hat{z})$ to $\mathcal{P}_{xz}$.
end for
Return: $\mathcal{P}_{xz}$.

The walkback procedure can be interpreted as a "wrapper" function which takes the corruption process q_ϕ and the reconstruction distribution p_θ, and then samples from a process $\mathcal{W}(z|x; q_\phi, p_\theta)$ which procedurally generates a distribution over $z \in \mathcal{Z}$ given any $x \in \mathcal{X}$, as shown in Alg. 1. For example, in the original GSN paper (Bengio et al., 2014), the reconstruction distribution $p_\theta(x|z)$ for a GSN which emulates Gibbs sampling in a Deep Boltzmann Machine was trained on pairs (x, z) sampled from the walkback process described in Alg 1. The returned set of pairs $\mathcal{P}_{xz}$ can be viewed as containing data $(x, \hat{z}) \sim \mathcal{W}(z|x; q_\phi, p_\theta)$, where $\mathcal{W}$ is specified procedurally rather than directly. The reconstruction distribution $p_\theta(x|z)$ is then trained to approximate the conditional $\mathcal{P}_{xz}(x|z)$ implicit in the pairs generated via Alg. 1.

3. Simple Generative Stochastic Networks

We define a "Simple GSN" as any GSN in which the corruption process renders z_t independent of z_{t-1} given x_{t-1}. Simple GSNs thus represent the minimal, direct extension of Generalized DAEs to corruption processes that may produce outputs in a different space from their inputs. Fig. 1(c) shows the structure of a simple GSN based on iteratively sampling from a walkback process $\mathcal{W}(z|x; q_\phi, p_\theta)$ and a reconstruction distribution $p_\theta(x|z)$.

The Simple GSN model is in fact quite general, and covers all GSNs trained with a walkback procedure. We now give versions of the theorems from (Bengio et al., 2013) modified for Simple GSNs, which show that training with enough data and with sufficiently powerful function approximators p_θ/q_ϕ produces a Markov chain whose asymp-

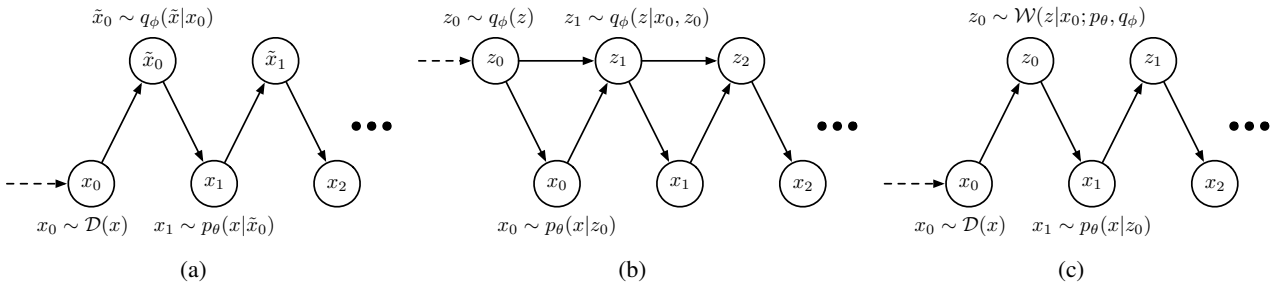


Figure 1. (a) shows how to construct the Markov chain associated with a Generalized DAE with reconstruction distribution p_θ, corruption process q_ϕ, and target distribution $\mathcal{D}$. (b) shows how to construct the Markov chain associated with a GSN with reconstruction distribution p_θ and corruption process q_ϕ. We overload notation and define $q_\phi(z)$ to be the distribution matching the stationary distribution of the GSN chain over z. (c) shows how to construct the Markov chain associated with a Simple GSN that uses a reconstruction distribution p_θ and a process $\mathcal{W}(z|x; q_\phi, p_\theta)$ formed by wrapping p_θ and a corruption process q_ϕ through the walkback procedure (see text for details).

totic distribution exists and matches the target distribution.

Theorem 1. *If $p_\theta(x|z)$ is a consistent estimator of the true conditional distribution $\mathcal{P}(x|z)$ and the transition operator $T_\theta(x_{t+1}|x_t)$ that samples z_t from $q_\phi(z_t|x_t)$ and x_{t+1} from $p_\theta(x_{t+1}|z_t)$ defines an ergodic Markov chain, then as the number of examples used to train $p_\theta(x|z)$ goes to infinity (i.e. as $p_\theta(x|z)$ converges to $\mathcal{P}(x|z)$), the asymptotic distribution of the Markov chain defined by T_θ converges to the target distribution $\mathcal{D}$.*

The proof is a direct translation of the proof for Theorem 1 in (Bengio et al., 2013), but with zs replacing $\tilde{x}$s. Briefly, drawing an initial x from $\mathcal{D}$ and then sampling alternately from $q_\phi(z|x)$ and $\mathcal{P}(x|z)$ is equivalent to sampling from a Gibbs chain for the joint distribution over (x_i, z_i) generated by repeatedly sampling $x_i \sim \mathcal{D}$ and $z_i \sim q_\phi(z|x_i)$. Ergodicity guarantees the existence of an asymptotic distribution for the Markov chain. Since $p_\theta(x|z)$ converges to the true $\mathcal{P}(x|z)$, the asymptotic distribution of the chain converges to the marginal distribution of x in the Gibbs chain, which is just $\mathcal{D}(x)$.

Corollary 2. *Let $\mathcal{X}$ be a set in which every pair of points is connected by a finite-length path contained in $\mathcal{X}$. Suppose that for each $x \in \mathcal{X}$ there exists a "shell" set $\mathcal{S}_x \subseteq \mathcal{X}$ such that all paths between x and any point in $\mathcal{X} \setminus \mathcal{S}_x$ pass through some point in $\mathcal{S}_x$ whose shortest path to x has length > 0. Suppose that $\forall x' \in \mathcal{S}_x \cup \{x\}, \exists z_{xx'}$ such that $q_\phi(z_{xx'}|x) > 0$ and $p_\theta(x'|z_{xx'}) > 0$. Then, the Markov chain with transition operator $T_\theta(x_{t+1}|x_t) = \sum_z p_\theta(x_{t+1}|z) q_\phi(z|x_t)$ is ergodic.*

Proof. The chain is aperiodic because the assumptions imply that $\forall x, \exists z_{xx}$ such that $q_\phi(z_{xx}|x) > 0$ and $p_\theta(x|z_{xx}) > 0$, so $T_\theta(x_{t+1} = x|x_t = x) > 0$. To show that the chain is irreducible, note that by assumption, $\forall x' \in \mathcal{S}_x, T(x_{t+1} = x'|x_t = x) > 0$. For any $x' \notin \mathcal{S}_x$, consider the shortest path from x to x' and note that $\exists y \in \mathcal{S}_x$ on this path such that the shortest path between x and y is > 0 and $T_\theta(x_{t+1} = y|x_t = x) > 0$. Hence, $T(x_{t+1} = x'|x_t = x) > 0$, as the path $x \rightarrow x'$ can be decomposed into a finite sequence of finite-length segments, each with non-zero transition probability. Because the chain is over a finite state space it is also positive recurrent. As the chain is aperiodic, irreducible, and positive recurrent, it is also ergodic. □

The restricted dependency structure of Simple GSNs lets us avoid some complications faced by the proofs in (Bengio et al., 2014). Our proof of Corollary 2 also avoids reliance on an ϵ ball, which does not work correctly in discrete or discontinuous spaces, in which paths starting at x with length $> \epsilon$ may contain no "segments" overlapping with the set of all paths starting at x with length $\leq \epsilon$.

Training $p_\theta(x|z)$ for any GSN using samples generated by walkback as described in Algorithm 1 corresponds to training a Simple GSN built around the reconstruction distribution p_θ and corruption process $\mathcal{W}(z|x; q_\phi, p_\theta)$. The key observation here is that the samples in $\mathcal{P}_{xz}$ generated by Algorithm 1 are obtained by relating *multiple* sampled latent variables $\hat{z}$ back to the *single* observable variable x given as input. In order to train p_θ to be consistent with the joint distribution over (x, z) pairs generated by the Markov chain constructed from p_θ and $q_\phi(z_t|x_{t-1}, z_{t-1})$, it would actually be necessary to train p_θ on pairs (x_{t-1}, z_t) generated by explicitly unrolling the chain. In Sec. 5 we present a mechanism based on Approximate Bayesian Computation that allows training p_θ directly on the pairs (x_{t-1}, z_t) generated by unrolling a GSN's Markov chain and applying backpropagation through time (BPTT).

Walkback can be viewed as an effective way to construct a more dispersed distribution over the latent space $\mathcal{Z}$ than

would be provided by the original corruption process q_ϕ. Though not explicitly stated in the existing work on GSNs, it seems that balancing between maximizing dispersion of the corruption process and the ease of modeling the reconstruction distribution p_θ plays a role for GSNs analogous to balancing between minimizing the KL divergence $\mathrm{KL}(q_\phi(z|x)||p(z))$ and maximizing the expected conditional log-likelihood $\mathbb{E}_{z\sim q_\phi(z|x)} \log p_\theta(x|z)$ when training a generative model $p_\theta(x)$ with variational methods, or balancing between following the "natural dynamics" of the system and optimizing reward in policy search (Peters et al., 2010; Todorov, 2009). The next section expands on this relation.

4. Variational Simple GSNs

We now develop a Simple GSN which can efficiently generate "locally-coherent" random walks along the manifold described by the target distribution $\mathcal{D}$, efficiently generate independent (approximate) samples from the target distribution $\mathcal{D}$, and efficiently evaluate a lower-bound on the log-likelihood assigned by the model to arbitrary inputs. We do this by replacing the denoising auto-encoders in existing examples of Generalized DAEs and GSNs with variational auto-encoders (Kingma & Welling, 2014; Rezende et al., 2014), while reinterpreting the two competing terms in the variational free-energy $\mathcal{F}$ (see Eq. 2) as representing an explicit trade-off between the dispersion of $q_\phi(z|x)$ and the ease of modeling $p_\theta(x|z)$.

Suppose that, in addition to $p_\theta(x|z)$ and $q_\phi(z|x)$, we also have access to a distribution $p_*(z)$ over $\mathcal{Z}$ (which could be learned or fixed a priori). Given these three distributions, we can define the *derived distribution* $p_\theta(x; p_*)$ such that $p_\theta(x; p_*) = \sum_z p_\theta(x|z)p_*(z)$, and the variational free-energy $\mathcal{F}(x; q_\phi, p_\theta, p_*)$, which provides an upper-bound on the negative log-likelihood of $x \in \mathcal{X}$ under $p_\theta(x; p_*)$:

$$\mathcal{F}(x; q_\phi, p_\theta, p_*) = \tag{1}$$
$$= -\sum_z [q_\phi(z|x) \log p_\theta(x|z)] + \mathrm{KL}(q_\phi(z|x)||p_*(z))$$
$$\geq -\log p_\theta(x; p_*) \tag{2}$$

A step-by-step derivation is provided in the Appendix.

Given $\mathcal{F}(x; q_\phi, p_\theta, p_*)$, we can maximize a lower-bound on the expected log-likelihood of samples $x \sim \mathcal{D}$ under the model $p_\theta(x; p_*)$ by minimizing:

$$\mathbb{E}_{x\sim\mathcal{D}} \quad \mathcal{F}(x; q_\phi, p_\theta, p_*) = \tag{3}$$
$$\mathbb{E}_{x\sim\mathcal{D}} \quad \left[\mathbb{E}_{z\sim q_\phi(z|x)} [-\log p_\theta(x|z)] + \mathrm{KL}(q_\phi(z|x)||p_*(z)) \right]$$

It is useful to compare this to the objective for Generalized DAEs, i.e. Eq. 4 in (Bengio et al., 2013):

$$\mathcal{L}(\theta) = \mathbb{E}_{x\sim\mathcal{D}, \tilde{x}\sim q_\phi(\tilde{z}|z)} [-\log p_\theta(x|\tilde{x}) + \lambda\Omega(\theta, x, \tilde{x})] \tag{4}$$

in which $\Omega(\theta, x, \tilde{x})$ is a regularization term for controlling the capacity of p_θ when the number of available samples $x \sim \mathcal{D}$ is finite. The basic training process for both Eq. 3 and 4 can be described as follows: draw a sample $x \sim \mathcal{D}$, apply a random corruption to get $z/\tilde{x} \sim q_\phi(\cdot|x)$, then adjust the parameters to reduce $-\log p_\theta(x|\cdot)$ and an added regularization term. Training with walkback simply changes the $\tilde{x} \sim q_\phi(\tilde{z}|z)$ in Eq. 4 to $\tilde{x} \sim \mathcal{W}(\tilde{x}|x; p_\theta, q_\phi)$.

We consider the objective in Eq. 3 from a GSN perspective and treat it as comprising two terms: a reconstruction error $-\log p_\theta(x|\cdot)$ and a dispersion maximization term $\mathrm{KL}(q_\phi(\cdot|x) \,||\, p_*)$. Based on a desire to keep q_ϕ well-dispersed for all $x \in \mathcal{X}$, and to keep the degree of the dispersion relatively consistent across $x \in \mathcal{X}$, we replace the basic KL term in Eq. 3 with the following:

$$\lambda \left(\mathrm{KL}(q_\phi(\cdot|x)||p_*) + \gamma([\mathrm{KL}(q_\phi(\cdot|x)||p_*) - \bar{K}]_+)^2 \right), \tag{5}$$

in which $\bar{K} = \mathbb{E}_{x\sim\mathcal{X}} \mathrm{KL}(q_\phi(\cdot|x)||p_*)$ and $[\cdot]_+$ indicates positive rectification, i.e. clamping all negative values to 0. This penalizes both the magnitude and variance of the per-example KL, while avoiding any pressure to increase KL. We make this modification to allow tighter control over the trade-off between reconstruction fidelity and dispersion of the corruption process. Although re-weighting the KL term in Eq. 3 may seem odd to those already familiar with variational methods, we emphasize that the free-energy $\mathcal{F}(x; q_\phi, p_\theta, p_*)$ from Eq. 2 still provides a valid upper-bound on $-\log p_\theta(x; p_*)$. The appendix provides further discussion of this variational free-energy.

5. Collaborative Generative Networks

In this section we take a step back and present a general method for shaping the distribution $\mathcal{G}$ produced by a generative model g_θ to be *practically* indistinguishable from a target distribution $\mathcal{D}$. In Section 6 we combine this method with variational Simple GSNs to directly train unrolled Markov chains. The general approach of estimating the parameters of g_θ to minimize some computable measure of dissimilarity between $\mathcal{G}$ and $\mathcal{D}$ is called Approximate Bayesian Computation. Examples of Approximate Bayesian Computation include spectral methods based on the "method of moments", which learn the parameters of g_θ so as to match some of the statistical moments of $\mathcal{G}$ to the corresponding moments observed in an empirical sample from $\mathcal{D}$, and more recent approaches based on minimizing the ability of some classifier to distinguish between $\mathcal{G}$ and $\mathcal{D}$ (Gutmann et al., 2014a;b; Goodfellow et al., 2014). Motivated by the recent empirical success of this latter approach in training deep generative models (Goodfellow et al., 2014), we develop a related approach which offers improved stability and a simpler proof of correctness.

We define an objective for jointly optimizing a *generator* function g_θ and a *guide* function f_ψ which shapes the distribution $\mathcal{G}$ produced by g_θ to match a target distribution $\mathcal{D}$. Our method can be interpreted as a collaboration between g_θ and f_ψ, in contrast with the adversarial approach presented in (Goodfellow et al., 2014). It is based on optimizing a term which encourages f_ψ to approximate the log-density-ratio $\log \frac{\mathcal{D}(x)}{\mathcal{G}(x)}$ for $x \in \mathcal{X}$ while also using f_ψ as a guidance signal for redistributing the mass emitted by g_θ. Our objective comprises two parts: one optimized by f_ψ and the other optimized by g_θ. We show that g_θ and f_ψ can simultaneously minimize their respective objectives if and only if $\forall x, \mathcal{G}(x) = \mathcal{D}(x)$ and $f_\psi(x) = \log \frac{\mathcal{D}(x)}{\mathcal{G}(x)} = 0$.

The objective $\mathcal{L}_f$ for f_ψ is the basic logistic regression loss for a binary classifier which assumes equal prior probability for the positive class $\mathcal{D}$ and the negative class $\mathcal{G}$, i.e.:

$$\mathcal{L}_f = \mathop{\mathbb{E}}_{x \sim \mathcal{D}} [b(f_\psi(x))] + \mathop{\mathbb{E}}_{x \sim \mathcal{G}} [b(-f_\psi(x))] \tag{6}$$

where $b(f) = \log(\exp(-f) + 1)$ is the binomial deviance loss. The objective $\mathcal{L}_g$ for g_θ is based on, e.g., a one-sided absolute value loss:

$$\mathcal{L}_g = \mathop{\mathbb{E}}_{x \sim \mathcal{G}} [|f_\psi(x)| \cdot \mathbb{I}[f_\psi(x) < 0]] \tag{7}$$

in which $\mathbb{I}[\cdot]$ is a binary indicator function.

Theorem 3. *The objectives $\mathcal{L}_f$ and $\mathcal{L}_g$ are simultaneously optimized with respect to f_ψ and g_θ if and only if $\forall x \in \mathcal{X}, \mathcal{G}(x) = \mathcal{D}(x)$ and $f_\psi(x) = \log \frac{\mathcal{D}(x)}{\mathcal{G}(x)} = 0$.*

Proof. By definition of $\mathcal{L}_f$, it is minimized w.r.t. f_ψ if and only if $\forall x,\ f_\psi(x) = \log \frac{\mathcal{D}(x)}{\mathcal{G}(x)}$ (Hastie et al., 2008). If $\mathcal{L}_f$ is minimized w.r.t. f_ψ then we know furthermore that either $\forall x,\ f_\psi(x) = \log \frac{\mathcal{D}(x)}{\mathcal{G}(x)} = 0$ or $\exists x$ s.t. $f_\psi(x) = \log \frac{\mathcal{D}(x)}{\mathcal{G}(x)} > 0$ and $\exists x'$ s.t. $f_\psi(x') = \log \frac{\mathcal{D}(x')}{\mathcal{G}(x')} < 0$. The former situation results in $\mathcal{L}_g = 0$. The latter situation results in $\mathcal{L}_g > 0$, because $|f_\psi(x)| \cdot \mathbb{I}[f_\psi(x) < 0] \geq 0$ with equality only when $f_\psi(x) \geq 0$. Thus, whenever $\mathcal{L}_f$ is optimized w.r.t. f_ψ, $\mathcal{L}_g$ can obtain its minimum possible value of 0 if and only if $\forall x,\ \mathcal{D}(x) = \mathcal{G}(x)$. □

Note that this proof works for any $\mathcal{L}_g$ which involves an expectation (w.r.t. $x \sim \mathcal{G}$) over a quantity which is everywhere non-negative and equal to 0 if and only if $f_\psi(x) \geq 0$. We leave a detailed investigation of the relative merits of the various possible $\mathcal{L}_g$ for future work.

The key characteristic that distinguishes our objective from (Goodfellow et al., 2014) is that, given a fixed guide function f_ψ, our objective for the generator g_θ pushes the mass in over-dense regions of $\mathcal{G}$ towards zero-contours of f_θ while leaving the mass in under-dense regions unmoved.

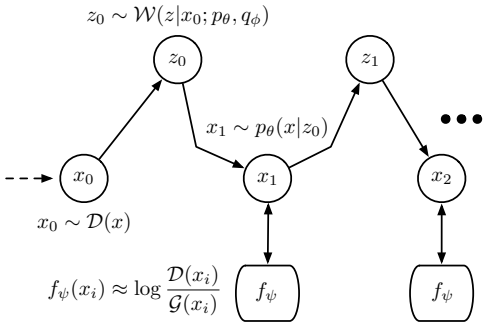


Figure 2. The VCG Loop: a self-looped variational auto-encoder whose asymptotic distribution is shaped by a guide function f_ψ using the collaborative mechanism described in Sec. 5.

In contrast, the adversarial objective in (Goodfellow et al., 2014) drives all mass emitted by g_θ towards local maxima of f_ψ. These local maxima will typically correspond to points in $\mathcal{X}$, while the zero-contours sought by our objective will correspond to regions in $\mathcal{X}$. We can also incorporate additional terms in $\mathcal{L}_g$, under the restriction that the added terms are minimized when $\forall x,\ \mathcal{D}(x) = \mathcal{G}(x)$. E.g. moment matching terms for matching the mean and covariance of $x \sim \mathcal{G}$ with those of $x \sim \mathcal{D}$ can be included to help avoid the occasional empirical "collapses" of $\mathcal{G}$ that were described in (Goodfellow et al., 2014).

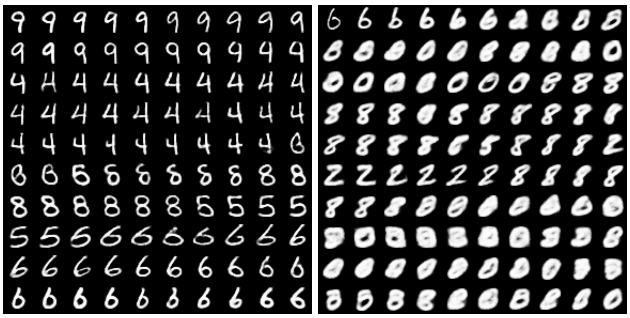
6. Generating Random Walks on Manifolds

We now combine the variational Simple GSNs from Sec. 4 with the collaborative mechanism from Sec. 5. Our goal is to directly train the Markov chain constructed by unrolling the variational Simple GSN to produce locally-contiguous walks along the manifold of the target distribution $\mathcal{D}$, and to have the asymptotic distribution of the chain approximate $\mathcal{D}$.

The collaborative mechanism described in Sec. 5 pairs a generator g_θ with a guide function f_ψ. For the generator, we propose using a variational Simple GSN, which we unroll into a Markov chain by initializing with a sample $x_0 \sim \mathcal{D}$ and then repeatedly generating $\{x_1, ..., x_t, ..., x_n\}$ by sampling $z_t \sim q_\phi(z|x_t)$ $x_{t+1} \sim p_\theta(x|z_t)$. In other words, we self-loop a variational auto-encoder by piping its output back into its input. The unrolled joint system is depicted in Fig. 2. In practice, we implement p_θ, q_ϕ and the guide function f_ψ using neural networks, whose specific architectures are described further in Sec. 7.

7. Experiments

We now present tests examining the behavior of our models on the MNIST and TFD datasets. We chose these datasets



(a) (b)

Figure 3. Comparing the chains generated by models learned with and without collaboratively-guided unrolling. The samples in (a) were generated by a corrupt-reconstruct pair q_ϕ/p_θ trained for 100k updates as a variational auto-encoder (VAE), and then 100k updates as a 6-step unrolled chain guided by a function f_ψ as described in Sec. 5. The samples in (b) were generated by a model with the same architecture and hyperparameters as in (a), but with 200k updates of VAE training. These chains are downsampled 5x.

to allow direct comparison with (Bengio et al., 2014) and (Goodfellow et al., 2014).

Our first tests with MNIST data examined the benefits of training using the unrolled collaborative mechanism in Fig. 2. We represented q_ϕ and p_θ using neural networks with two hidden layers of 1500 rectified-linear units and we set the latent space $\mathcal{Z}$ to $\mathbb{R}^{64}$. We used a Gaussian with identity covariance for the prior distribution $p_*(x)$. The output layer of q_ϕ produced two vectors in $\mathbb{R}^{64}$, one representing the mean of a Gaussian distribution over $\mathcal{Z}$ and the other representing the element-wise log-variances of the distribution. Given this q_ϕ/p_*, $\mathrm{KL}(q_\phi(z|x)||p_*)$ was easy to compute analytically, and its gradients with respect to ϕ were readily available. We interpreted $p_\theta(x|z)$ as a factored Bernoulli distribution, with Rao-Blackwellisation over the possible binarizations of each x. I.e., the output layer of p_θ produced a vector in $\mathbb{R}^{784}$, which was then passed through a sigmoid to get $\hat{x}$. We minimized $-\log p_\theta(x|z) = -\mathrm{sum}(x \odot \log \hat{x} + (1-x) \odot \log(1-\hat{x}))$, where $\odot$ denotes element-wise multiplication and we sum the vector entries. The gradients of $-\log p_\theta(x|z)$ w.r.t. θ were directly available, and we backpropped through sampling $z \sim q_\phi(z|x)$ to get gradients w.r.t. ϕ using the techniques in (Kingma & Welling, 2014; Rezende et al., 2014).

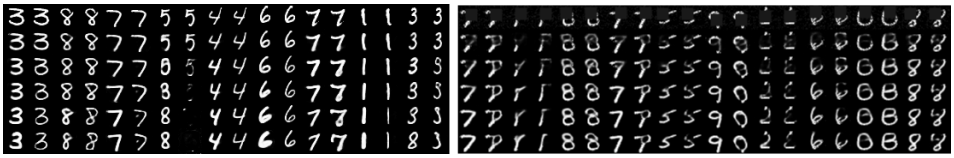
We used a Maxout network (Goodfellow et al., 2013) with two hidden layers of 1200 units in 300 groups of 4 for the guide function f_ψ. For $\mathcal{L}_g$ in Eq. 7, we used a half-rectified elastic-net (Zou & Hastie, 2005), with the linear and quadratic parts weighted equally. We unrolled our chains for 6 steps. The distributions $\mathcal{D}$ and $\mathcal{G}$ for training f_ψ according to Eq. 6 were given by the MNIST training set and the x_i emitted by the unrolled chains. We passed gradients through the unrolled computation graph via BPTT.

We performed model updates using gradients computed from mini-batches of 100 distinct examples from the training set, each of which was passed through the model for 5 samples from $q_\phi(z|x)$. We trained using plain SGD. We pre-trained p_θ and q_ϕ as a variational auto-encoder (VAE) for 100k updates by running the model in Fig. 1(c) out to x_1. After 100k VAE updates we "forked" the model into a "multi-step guided" model and a "one-step VAE" model and performed a further 100k updates to each of the now-independent models. We implemented our models in Python using the THEANO library (Bergstra et al., 2010). Code implementing the models described in this paper is available online at: `github.com/Philip-Bachman/ICML-2015`. The code provides full details on learning rates and other hyper-parameters.

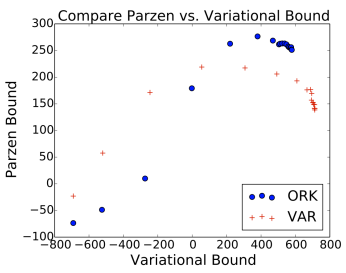
Fig. 3 illustrates the significantly improved long-run sampling behavior of chains trained with unrolling. The chains in Fig. 3 start at the top left and run from left-to-right and top-to-bottom. The true samples from $\mathcal{D}$ used to initialize the chains are in the top-left corners of (a)/(b). We downsampled the samples emitted by these chains 5x for this figure. Training with unrolling and collaborative guidance allows the chain to continually generate clean samples while exhibiting rapid mixing between the modes of the target distribution. Without explicit unrolling during training, the chain can only perform a few steps before degrading into samples that are well-separated from the target distribution. Qualitatively, the samples generated by the model trained with collaboratively-guided unrolling compare favourably with those presented for GSNs in (Bengio et al., 2014).

For our second tests with the MNIST images, we used networks with roughly the same structure as in our first tests but we set $\mathcal{Z}$ to $\mathbb{R}^{50}$, we used 1000 units in each of the hidden layers in q_ϕ/p_θ, and we used a Gaussian reconstruction distribution $p_\theta(x|z)$. To effect this final change, we interpreted the vector produced by the output layer of p_θ as the mean of a Gaussian distribution over $\mathcal{X}$ and we shared a single "bias" parameter across all z to model the element-wise log-variance of $p_\theta(x|z)$. We thus modeled the target distribution using an infinite mixture of isotropic Gaussians, with the mixture weights of each Gaussian fixed a priori and with their individual locations and shared scale adjusted to match the training data.

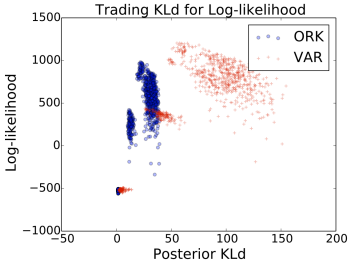
For Fig. 4(a)-(d), we trained a pair of models. We trained the first model with $\lambda = 4$ and $\gamma = 0.1$ in Eq. 5. We refer to this model as ORK, for over-regularized KL. We trained the second model to minimize the basic free energy in Eq. 2. We refer to this model as VAR. As in our first MNIST experiments, we initialized each model with pre-training by running the model in Fig. 1(c) a single step. We performed 80k pre-training updates using mini-batches



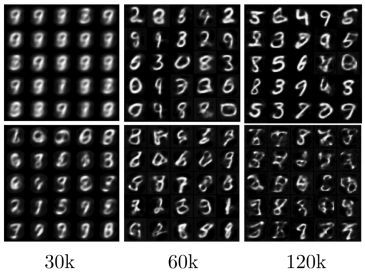
(a)



(b)

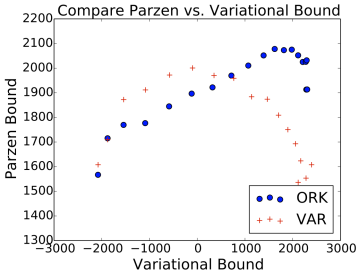


(c)

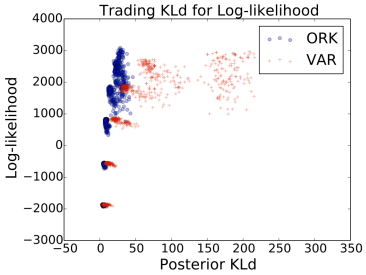


(d)

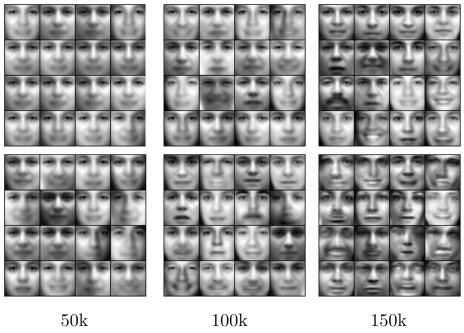
(e)



(f)



(g)



(h)

Figure 4. MNIST and TFD results. (a) compares the behavior of chains generated by models trained with over-regularized posterior KL divergence (ORK) and chains generated by models trained to minimize the standard variational free energy (VAR). The top row gives the digit image with which each chain was initialized, and the left/right chain in each pair sharing the same initialization represent chains generated by the ORK/VAR models. For the left block of examples the chains were allowed to run "freely" after initialization, and we show every 5th sample generated by the chains. For the right block the chains were run under "partial control", wherein some subset of the pixels were held fixed at their initial values while the remaining pixels were occluded at initialization and "painted in" by the model over multiple steps. We show every 2nd generated sample. (b) shows how the ORK and VAR models perform with respect to bounds on the validation set log-likelihood provided by the Gaussian Parzen density estimator described in (Breuleux et al., 2011) and the variational free energy described in Eq. 2. We computed these values after every 10k parameter updates. (c) shows the joint distribution over the reconstruction and posterior KL terms (i.e. $\log p_\theta(x|z)$ and $\mathrm{KL}(q_\phi(z|x)||p_*(z))$) in Eq. 2 measured at several points during the training of the ORK and VAR models. We computed these values after every 30k parameter updates by averaging over 10 passes through the system comprising q_ϕ and p_θ for each of 150 examples selected at random from a validation set. (d) shows independent samples from the ORK (top row) and VAR (bottom row) models after 30k, 60k, and 120k parameter updates. (e)-(h) provide results for TFD analogous to those in (a)-(d) for MNIST, with differences in the tests/models detailed in the main text.

and SGD updates as described for our first experiments. We then performed another 120k updates by unrolling the chains for 6 steps following the graph in Fig. 2. The guide function f_ψ was trained as in our first experiments.

Fig. 4 illustrates interesting behaviors of the ORK and VAR models. We found that the ORK model with strong regularization on $\mathrm{KL}(q_\phi(z|x)||p_*(z))$ was able to significantly out-perform the VAR model according to the Gaussian Parzen density estimator (GPDE) test described in (Breuleux et al., 2011)[1]. On the test set, the best ORK model scored 265, which compares favorably to the 214 in (Bengio et al., 2014) and the 225 in (Goodfellow et al., 2014). The best VAR model scored 220. The peak performance by this metric occurred much earlier in training for the VAR model than for the ORK model. Using the same network architecture, but with λ in Eq. 5 increased to 24, our approach achieved a score of 330 on the GPDE test.

Interestingly, the GPDE log-likelihood bound began to decrease rapidly beyond a certain point in training. However, the variational bound continued to increase. Qualitatively, this behavior is clearly reflected in the samples shown in Fig. 4(d), which we drew directly from the models by sampling from their priors $p_*(x)$. Samples generated from the ORK model remain reasonable throughout training, but eventually suffer on the GPDE bound due to excess "sharpness". Samples drawn from the VAE model after 150k updates, when the VAE model significantly outperforms the ORK model in terms of the variational bound, are hardly recognizable as handwritten digits. In effect, the model is concentrating its posterior mass, as given by $q_\phi(z|x)$, on increasingly small regions of $\mathcal{Z}$, in exchange for significant reductions in the reconstruction cost $-\log p_\theta(x|z)$. This seems to lead to most of the mass of $p_*(z)$ falling on zs which have little or no mass under any of the $q_\phi(z|x)$. By forcing the $q_\phi(z|x)$ to be more dispersed over $\mathcal{Z}$, our added KL terms in Eq. 5 help mitigate this issue, albeit at the cost of less precise reconstruction of any particular digit. The scatter plots in Fig. 4 illustrate the evolution of the GPDE/variational bounds over the course of training and the trade-off between reconstruction cost and posterior KL that is obtained by the ORK and VAR models.

We performed analogous tests with the TFD dataset, which comprises 48x48 grayscale images of frontal faces in various expressions. We made a few changes from the second set of MNIST tests. We expanded the hidden layers of the networks representing q_ϕ/p_θ to 2000 rectified-linear units each and we expanded the latent space $\mathcal{Z}$ to $\mathbb{R}^{100}$. We pre-processed the images to have pixel intensities in the range $[0...1]$, as in (Bengio et al., 2014; Goodfellow et al., 2014). We extended the pre-training phase to 150k updates and the unrolled, collaboratively-guided phase was reduced to 60k updates. Interestingly, in these tests the ORK model did not suffer significantly in terms of the variational bound, while achieving dramatically improved performance on the GPDE bound. For comparison, best previous results on the GPDE bound for this dataset are 2050 (Goodfellow et al., 2014) and 2110 (Bengio et al., 2013). Our ORK model scored 2060 on the test set using a GPDE variance selected on the validation set. When we increased λ for the ORK model from 5 to 30, the GPDE score increased to 2130.

[1]Briefly, this test approximates the distribution of a generative model by drawing 10k samples from the model and then using those samples as the mixture means for a uniformly-weighted mixture of 10k Gaussians, with a shared isotropic variance selected for the mixture components based on a validation set

8. Discussion

We presented an approach for learning generative models belonging to a simple subset of GSNs, using variational auto-encoders as building blocks. We generated Markov chains by looping the output of these auto encoders back into the input, and trained them to generate random walks along a target manifold, based on feedback from a guide function trained to discriminate between samples emitted by the chain and samples drawn from the data manifold. A key conceptual contribution of our approach is that we run the generative process as an unrolled Markov chain according to its a natural dynamics, i.e. the same way we want to run it "in the wild", and then correct differences between exhibited and desired behavior by providing direct feedback about their magnitude and location (instead of trying to force the behavior in some way during the generation process). The experimental evaluation demonstrates that this direct approach is beneficial in practice.

In the long run, we believe it will be interesting to focus on interpreting our method from a reinforcement learning point of view. In this case, the "ease of modeling" the posterior distribution $p(x|z)$ can be viewed as a reward to be maximized, and may be easily replaced or augmented with other sources of reward. The current approach of using back propagation through time for the training could also be replaced by more efficient methods based on eligibility traces. While we only considered generating random walks over manifolds in this paper, in future work we would like to apply our approach to modeling distributions over observed trajectories on manifolds, e.g., as seen in speech, video, motion capture, and other sequential data.

References

Bengio, Yoshua, Yao, Li, Alain, Guillaume, and Vincent, Pascal. Generalized denoising auto-encoders as generative models. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

Bengio, Yoshua, Thibodeau-Laufer, Éric, Alain, Guillaume, and Yosinski, Jason. Deep generative stochastic networks trainable by backprop. *arXiv:1306.1091v5 [cs.LG]*, 2014.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. Theano: A cpu and gpu math expression compiler. In *Python for Scientific Computing Conference (SciPy)*, 2010.

Breuleux, Olivier, Bengio, Yoshua, and Vincent, Pascal. Quickly generating representative samples from an rbm-derived process. *Neural Computation*, 23(8):2053–2073, 2011.

Goodfellow, Ian J, Warde-Farley, David, Mirza, Mehdi, Courville, Aaron, and Bengio, Yoshua. Maxout networks. In *International Conference on Machine Learning (ICML)*, 2013.

Goodfellow, Ian J, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

Gutmann, Michael U, Dutta, Ritabrata, Kaski, Samuel, and Corander, Jukka. Classifier abc. In *MCMSki IV (posters)*, 2014a.

Gutmann, Michael U, Dutta, Ritabrata, Kaski, Samuel, and Corander, Jukka. Likelihood-free inference via classification. In *arXiv:1407.4981v1 [stat.CO]*, 2014b.

Hastie, Trevor, Friedman, Jerome, and Tibshirani, Robert. *Elements of Statistical Learning II*. 2008.

Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv:1412.6980v2 [cs.LG]*, 2015.

Kingma, Diederik P and Welling, Max. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

Levine, Sergey and Koltun, Vladen. Guided policy search. In *International Conference on Machine Learning (ICML)*, 2013.

Levine, Sergey, Wagener, Nolan, and Abbeel, Pieter. Learning contact-rich manipulation skills with guided policy search. In *ICRA*, 2015.

Mnih, Andriy and Gregor, Karol. Neural variational inference and learning. In *International Conference on Machine Learning (ICML)*, 2014.

Peters, Jan, Mulling, Karen, and Altun, Yasemin. Relative entropy policy search. In *AAAI*, 2010.

Rezende, Danilo, Mohamed, Shakir, and Wierstra, Daan. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*, 2014.

Todorov, Emanuel. Efficient computation of optimal action. *Proceedings of the National Academy of Science (PNAS)*, 106(18):11478–11483, 2009.

Zou, Hui and Hastie, Trevor. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, B.*, 67(2):301–320, 2005.