Abstract

A central problem in machine learning involves modeling complex data-sets using highly flexible families of probability distributions in which learning, sampling, inference, and evaluation are still analytically or computationally tractable. Here, we develop an approach that simultaneously achieves both flexibility and tractability. The essential idea, inspired by non-equilibrium statistical physics, is to systematically and slowly destroy structure in a data distribution through an iterative forward diffusion process. We then learn a reverse diffusion process that restores structure in data, yielding a highly flexible and tractable generative model of the data. This approach allows us to rapidly learn, sample from, and evaluate probabilities in deep generative models with thousands of layers or time steps, as well as to compute conditional and posterior probabilities under the learned model. We additionally release an open source reference implementation of the algorithm.

1. Introduction

Historically, probabilistic models suffer from a tradeoff between two conflicting objectives: tractability and flexibility. Models that are tractable can be analytically evaluated and easily fit to data (e.g. a Gaussian or Laplace). However, these models are unable to aptly describe structure in rich datasets. On the other hand, models that are flexible can be molded to fit structure in arbitrary data. For example, we can define models in terms of any (non-negative) function \( \phi(x) \) yielding the flexible distribution \( p(x) = \frac{\phi(x)}{Z} \), where \( Z \) is a normalization constant. However, computing this normalization constant is generally intractable. Evaluating, training, or drawing samples from such flexible models typically requires a very expensive Monte Carlo process.

A variety of analytic approximations exist which ameliorate, but do not remove, this tradeoff—for instance mean field theory and its expansions (T, 1982; Tanaka, 1998), variational Bayes (Jordan et al., 1999), contrastive divergence (Welling & Hinton, 2002; Hinton, 2002), minimum probability flow (Sohl-Dickstein et al., 2011b;a), minimum KL contraction (Lyu, 2011), proper scoring rules (Gneiting & Raftery, 2007; Parry et al., 2012), score matching (Hyvärinen, 2005), pseudolikelihood (Besag, 1975), loopy belief propagation (Murphy et al., 1999), and many, many more. Non-parametric methods (Gershman & Blei, 2012) can also be very effective.\(^1\)

1. Diffusion probabilistic models

We present a novel way to define probabilistic models that allows:

1. extreme flexibility in model structure,
2. exact sampling.

\(^1\)Non-parametric methods can be seen as transitioning smoothly between tractable and flexible models. For instance, a non-parametric Gaussian mixture model will represent a small amount of data using a single Gaussian, but may represent infinite data as a mixture of an infinite number of Gaussians.
3. easy multiplication with other distributions, e.g. in order to compute a posterior, and
4. the model log likelihood, and the probability of individual states, to be cheaply evaluated.

Our method uses a Markov chain to gradually convert one distribution into another, an idea used in non-equilibrium statistical physics (Jarzynski, 1997) and sequential Monte Carlo (Neal, 2001). We build a generative Markov chain which converts a simple known distribution (e.g. a Gaussian) into a target (data) distribution using a diffusion process. Rather than use this Markov chain to approximately evaluate a model which has been otherwise defined, we explicitly define the probabilistic model as the endpoint of the Markov chain. Since each step in the diffusion chain has an analytically evaluable probability, the full chain can also be analytically evaluated.

Learning in this framework involves estimating small perturbations to a diffusion process. Estimating small perturbations is more tractable than explicitly describing the full distribution with a single, non-analytically-normalizable, potential function. Furthermore, since a diffusion process exists for any smooth target distribution, this method can capture data distributions of arbitrary form.

We demonstrate the utility of these diffusion probabilistic models by training high log likelihood models for a two-dimensional swiss roll, binary sequence, handwritten digit (MNIST), and several natural image (CIFAR-10, bark, and dead leaves) datasets.

1.2. Relationship to other work

The wake-sleep algorithm (Hinton, 1995; Dayan et al., 1995) introduced the idea of training inference and generative probabilistic models against each other. This approach remained largely unexplored for nearly two decades, though with some exceptions (Smichisescu et al., 2006; Kavukcuoglu et al., 2010). There has been a recent explosion of work developing this idea. In (Kingma & Welling, 2013; Gregor et al., 2013; Rezende et al., 2014; Ozair & Bengio, 2014) variational learning and inference algorithms were developed which allow a flexible generative model and posterior distribution over latent variables to be directly trained against each other.

The variational bound in these papers is similar to the one used in our training objective and in the earlier work of (Smichisescu et al., 2006). However, our motivation and model form are both quite different, and the present work retains the following differences and advantages relative to these techniques:

1. We develop our framework using ideas from physics, quasi-static processes, and annealed importance sampling rather than from variational Bayesian methods.
2. We show how to easily multiply the learned distribution with another probability distribution (e.g. with a conditional distribution in order to compute a posterior)
3. We address the difficulty that training the inference model can prove particularly challenging in variational inference methods, due to the asymmetry in the objective between the inference and generative models. We restrict the forward (inference) process to a simple functional form, in such a way that the reverse (generative) process will have the same functional form.
4. We train models with thousands of layers (or time steps), rather than only a handful of layers.
5. We provide upper and lower bounds on the entropy production in each layer (or time step)

There are a number of related techniques for training probabilistic models (summarized below) that develop highly flexible forms for generative models, train stochastic trajectories, or learn the reversal of a Bayesian network. Reweighted wake-sleep (Bornschein & Bengio, 2015) develops extensions and improved learning rules for the original wake-sleep algorithm. Generative stochastic networks (Bengio & Thibodeau-Lafer, 2013; Yao et al., 2014) train a Markov kernel to match its equilibrium distribution to the data distribution. Neural autoregressive distribution estimators (Larochelle & Murray, 2011) (and their recurrent (Uria et al., 2013a) and deep (Uria et al., 2013b) extensions) decompose a joint distribution into a sequence of tractable conditional distributions over each dimension. Adversarial networks (Goodfellow et al., 2014) train a generative model against a classifier which attempts to distinguish generated samples from true data. A similar objective in (Schmidhuber, 1992) learns a two-way mapping to a representation with marginally independent units. In (Rippel & Adams, 2013; Dinh et al., 2014) bijective deterministic maps are learned to a latent representation with a simple factorial density function. In (Stuhlmuller et al., 2013) stochastic inverses are learned for Bayesian networks. Mixtures of conditional Gaussian scale mixtures (MCGSMs) (Theis et al., 2012) describe a dataset using Gaussian scale mixtures, with parameters which depend on a sequence of causal neighborhoods. There is additionally significant work learning flexible generative mappings from simple latent distributions to data distributions – early examples including (MacKay, 1995) where neural networks are introduced as generative models, and (Bishop et al., 1998) where a stochastic manifold mapping is learned from a latent space to the data space. We will compare experimentally against adversarial networks and MCGSMs.

Related ideas from physics include the Jarzynski equality (Jarzynski, 1997), known in machine learning as An-
Deep Unsupervised Learning using Nonequilibrium Thermodynamics

2. Algorithm

Our goal is to define a forward (or inference) diffusion process which converts any complex data distribution into a simple, tractable, distribution, and then learn a finite-time reversal of this diffusion process which defines our generative model distribution (See Figure 1). We first describe the forward, inference diffusion process. We then show how the reverse, generative diffusion process can be trained and used to evaluate probabilities. We also derive entropy bounds for the reverse process, and show how the learned distributions can be multiplied by any second distribution (e.g. as would be done to compute a posterior when inpainting or denoising an image).

2.1. Forward Trajectory

We label the data distribution \( q (x^{(0)} \cdot T) \). The data distribution is gradually converted into a well behaved (analytically tractable) distribution \( \pi (y) \) by repeated application of a Markov diffusion kernel \( T_\pi (y|y'; \beta) \) for \( \pi (y) \), where \( \beta \) is the diffusion rate.

\[
\pi (y) = \int dy' T_\pi (y|y'; \beta) \pi (y') \quad (1)
\]

\[
q (x^{(t)}|x^{(t-1)}) = T_\pi (x^{(t)}|x^{(t-1)}; \beta_t) \quad (2)
\]
Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Figure 2. Binary sequence learning via binomial diffusion. A binomial diffusion model was trained on binary ‘heartbeat’ data, where a pulse occurs every 5th bin. Generated samples (left) are identical to the training data. The sampling procedure consists of initialization at independent binomial noise (right), which is then transformed into the data distribution by a binomial diffusion process, with trained bit flip probabilities. Each row contains an independent sample. For ease of visualization, all samples have been shifted so that a pulse occurs in the first column. In the raw sequence data, the first pulse is uniformly distributed over the first five bins.

Figure 3. The proposed framework trained on the CIFAR-10 (Krizhevsky & Hinton, 2009) dataset. (a) Example training data. (b) Random samples generated by the diffusion model.

The forward trajectory, corresponding to starting at the data distribution and performing $T$ steps of diffusion, is thus

$$q(x^{(0...T)}) = q(x^{(0)}) \prod_{t=1}^{T} q(x^{(t)}|x^{(t-1)})$$

For the experiments shown below, $q(x^{(t)}|x^{(t-1)})$ corresponds to either Gaussian diffusion into a Gaussian distribution with identity-covariance, or binomial diffusion into an independent binomial distribution. Table C.1 gives the diffusion kernels for both Gaussian and binomial distributions.

2.2. Reverse Trajectory

The generative distribution will be trained to describe the same trajectory, but in reverse,

$$p(x^{(T)}) = \pi(x^{(T)})$$

$$p(x^{(0...T)}) = p(x^{(T)}) \prod_{t=1}^{T} p(x^{(t-1)}|x^{(t)})$$

For both Gaussian and binomial diffusion, for continuous diffusion (limit of small step size $\beta$) the reversal of the diffusion process has the identical functional form as the forward process (Feller, 1949). Since $q(x^{(t)}|x^{(t-1)})$ is a Gaussian (binomial) distribution, and if $\beta_t$ is small, then $q(x^{(t-1)}|x^{(t)})$ is also a Gaussian (binomial) distribution. The longer the trajectory the smaller the diffusion rate $\beta$ can be made.

During learning only the mean and covariance for a Gaussian diffusion kernel, or the bit flip probability for a binomial kernel, need be estimated. As shown in Table C.1, $f_{\mu}(x^{(t)}, t)$ and $f_{\Sigma}(x^{(t)}, t)$ are functions defining the mean and covariance of the reverse Markov transitions for a Gaussian, and $f_b(x^{(t)}, t)$ is a function providing the bit flip probability for a binomial distribution. The computational cost of running this algorithm is the cost of the these functions, times the number of time-steps. For all results in this paper, multi-layer perceptrons are used to define these functions. A wide range of regression or function fitting techniques would be applicable however, including non-parametric methods.

2.3. Model Probability

The probability the generative model assigns to the data is

$$p(x^{(0)}) = \int dx^{(1...T)} p(x^{(0...T)})$$
Naively this integral is intractable – but taking a cue from annealed importance sampling and the Jarzynski equality, we instead evaluate the relative probability of the forward and reverse trajectories, averaged over forward trajectories,

\[
p(x^{(0)}) = \int d\mathbf{x}^{(1:T)} p(\mathbf{x}^{(0:T)}) \frac{q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)})}{q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)})} \tag{7}
\]

\[
= \int d\mathbf{x}^{(1:T)} q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)}) \frac{p(\mathbf{x}^{(0:T)})}{q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)})} \tag{8}
\]

\[
= \int d\mathbf{x}^{(1:T)} q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)}) \cdot p(\mathbf{x}^{(T)}) \prod_{i=1}^{T} \frac{p(\mathbf{x}^{(i-1)}|\mathbf{x}^{(i)})}{q(\mathbf{x}^{(i)}|\mathbf{x}^{(i-1)})} \tag{9}
\]

This can be evaluated rapidly by averaging over samples from the forward trajectory \(q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)})\). For infinitesimal \(\beta\) the forward and reverse distribution over trajectories can be made identical (see Section 2.2). If they are identical then only a single sample from \(q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)})\) is required to exactly evaluate the above integral, as can be seen by substitution. This corresponds to the case of a quasi-static process in statistical physics (Spinney & Ford, 2013; Jarzynski, 2011).

2.4. Training

Training amounts to maximizing the model log likelihood,

\[
L = \int d\mathbf{x}^{(0)} q(\mathbf{x}^{(0)}) \log p(\mathbf{x}^{(0)}) \tag{10}
\]

\[
= \int d\mathbf{x}^{(0)} q(\mathbf{x}^{(0)}) \cdot \log \left[ \int d\mathbf{x}^{(1:T)} q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)}) \cdot p(\mathbf{x}^{(T)}) \prod_{i=1}^{T} \frac{p(\mathbf{x}^{(i-1)}|\mathbf{x}^{(i)})}{q(\mathbf{x}^{(i)}|\mathbf{x}^{(i-1)})} \right] \tag{11}
\]

which has a lower bound provided by Jensen’s inequality,

\[
L \geq \int d\mathbf{x}^{(0:T)} q(\mathbf{x}^{(0:T)}).
\]

\[
= \log p(\mathbf{x}^{(T)}) \prod_{i=1}^{T} \frac{p(\mathbf{x}^{(i-1)}|\mathbf{x}^{(i)})}{q(\mathbf{x}^{(i)}|\mathbf{x}^{(i-1)})} \tag{12}
\]

As described in Appendix B, for our diffusion trajectories this reduces to,

\[
L \geq K \tag{13}
\]

\[
K = -\sum_{i=2}^{T} \int d\mathbf{x}^{(0)} d\mathbf{x}^{(i)} q(\mathbf{x}^{(0)}, \mathbf{x}^{(i)}) .
\]

\[
D_{KL} [ q(\mathbf{x}^{(i-1)}|\mathbf{x}^{(i)}, \mathbf{x}^{(0)}) ] p(\mathbf{x}^{(i-1)}|\mathbf{x}^{(i)})
\]

\[
+ H_q(\mathbf{x}^{(T)}|\mathbf{x}^{(1)}) - H_q(\mathbf{x}^{(1)}|\mathbf{x}^{(0)}) - H_p(\mathbf{x}^{(T)}), \tag{14}
\]

where the entropies and KL divergences can be analytically computed. The derivation of this bound parallels the derivation of the log likelihood bound in variational Bayesian methods.

As in Section 2.3 if the forward and reverse trajectories are identical, corresponding to a quasi-static process, then the inequality in Equation 13 becomes an equality.

Training consists of finding the reverse Markov transitions which maximize this lower bound on the log likelihood,

\[
\hat{\rho}(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)}) = \arg\max_{\rho(\mathbf{x}^{(t-1)}|\mathbf{x}^{(t)})} K. \tag{15}
\]

The specific targets of estimation for Gaussian and binomial diffusion are given in Table C.1.

Thus, the task of estimating a probability distribution has been reduced to the task of performing regression on the functions which set the mean and covariance of a sequence of Gaussians (or set the state flip probability for a sequence of Bernoulli trials).

2.4.1. Setting the Diffusion Rate \(\beta_t\)

The choice of \(\beta_t\) in the forward trajectory is important for the performance of the trained model. In AIS, the right schedule of intermediate distributions can greatly improve the accuracy of the log partition function estimate (Grosse et al., 2013). In thermodynamics the schedule taken when moving between equilibrium distributions determines how much free energy is lost (Spinney & Ford, 2013; Jarzynski, 2011).

In the case of Gaussian diffusion, we learn\(^2\) the forward diffusion schedule \(\beta_{2:T}\) by gradient ascent on \(K\). The variance \(\beta_1\) of the first step is fixed to a small constant to prevent overfitting. The dependence of samples from \(q(\mathbf{x}^{(1:T)}|\mathbf{x}^{(0)})\) on \(\beta_{1:T}\) is made explicit by using “frozen noise” – as in (Kingma & Welling, 2013) the noise is treated as an additional auxiliary variable, and held constant while computing partial derivatives of \(K\) with respect to the parameters.

For binomial diffusion, the discrete state space makes gradient ascent with frozen noise impossible. We instead choose the forward diffusion schedule \(\beta_{1:T}\) to erase a constant fraction \(\frac{1}{T}\) of the original signal per diffusion step, yielding a diffusion rate of \(\beta_t = (T - t + 1)^{-1}\).

2.5. Multiplying Distributions, and Computing Posteriors

Tasks such as computing a posterior in order to do signal denoising or inference of missing values requires multipli-

\(^2\)Recent experiments suggest that it is just as effective to instead use the same fixed \(\beta_t\) schedule as for binomial diffusion.
culation of the model distribution $p(x^{(0)})$ with a second distribution, or bounded positive function, $r(x^{(0)})$, producing a new distribution $\tilde{p}(x^{(0)}) \propto p(x^{(0)}) r(x^{(0)})$.

Multiplying distributions is costly and difficult for many techniques, including variational autoencoders, GSNS, NADEs, and most graphical models. However, under a diffusion model it is straightforward, since the second distribution can be treated either as a small perturbation to each step in the diffusion process, or often exactly multiplied into each diffusion step. Figure 5 demonstrates the use of a diffusion model to perform inpainting of a natural image.

The following sections describe how to multiply distributions in the context of diffusion probabilistic models.

2.5.1. Modified Marginal Distributions

First, in order to compute $\tilde{p}(x^{(0)})$, we multiply each of the intermediate distributions by a corresponding function $r(x^{(t)})$. We use a tilde above a distribution or Markov transition to denote that it belongs to a trajectory that has been modified in this way. $\tilde{q}(x^{(0:1:7)})$ is the modified forward trajectory, which starts at the distribution $\tilde{q}(x^{(0)}) = \frac{1}{Z_t} q(x^{(0)}) r(x^{(0)})$ and proceeds through the sequence of intermediate distributions

$$
\tilde{q}(x^{(t)}) = \frac{1}{Z_t} q(x^{(t)}) r(x^{(t)}),
$$

where $Z_t$ is the normalizing constant for the $t$th intermediate distribution.

2.5.2. Modified Conditional Distributions

Next, writing the relationship between the forward and reverse conditional distributions demonstrates how multiplying each intermediate distribution by $r(x^{(t)})$ changes the Markov diffusion chain. By Bayes’ rule the forward chain presented in Section 2.1 satisfies

$$
q(x^{(t+1)} | x^{(t)}) q(x^{(t)}) = q(x^{(t)} | x^{(t+1)}) q(x^{(t+1)}).
$$

The new chain must instead satisfy

$$
\tilde{q}(x^{(t+1)} | x^{(t)}) \tilde{q}(x^{(t)}) = \tilde{q}(x^{(t)} | x^{(t+1)}) \tilde{q}(x^{(t+1)}).
$$

As derived in Appendix C, one way to choose a new Markov chain which satisfies Equation 18 is to set

$$
\tilde{q}(x^{(t+1)} | x^{(t)}) \propto q(x^{(t+1)} | x^{(t)}) r(x^{(t+1)}),
\tilde{q}(x^{(t)} | x^{(t+1)}) \propto q(x^{(t)} | x^{(t+1)}) r(x^{(t)}).
$$

So that $\tilde{p}(x^{(t)} | x^{(t+1)})$ corresponds to $\tilde{q}(x^{(t)} | x^{(t+1)})$, $p(x^{(t)} | x^{(t+1)})$ is modified in the corresponding fashion,

$$
\tilde{p}(x^{(t)} | x^{(t+1)}) \propto p(x^{(t)} | x^{(t+1)}) r(x^{(t)}).
$$

2.5.3. Applying $r(x^{(t)})$

If $r(x^{(t)})$ is sufficiently smooth, then it can be treated as a small perturbation to the reverse diffusion kernel $p(x^{(t)} | x^{(t+1)})$. In this case $\tilde{p}(x^{(t)} | x^{(t+1)})$ will have an identical functional form to $p(x^{(t)} | x^{(t+1)})$, but with perturbed mean and covariance for the Gaussian kernel, or with perturbed flip rate for the binomial kernel. The perturbed diffusion kernels are given in Table C.1.

If $r(x^{(t)})$ can be multiplied with a Gaussian (or binomial) distribution in closed form, then it can be directly multiplied with the reverse diffusion kernel $p(x^{(t)} | x^{(t+1)})$ in closed form, and need not be treated as a perturbation. This
Deep Unsupervised Learning using Nonequilibrium Thermodynamics

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$K$</th>
<th>$K - L_{null}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swiss Roll</td>
<td>2.35 bits</td>
<td>6.45 bits</td>
</tr>
<tr>
<td>Binary Heartbeat</td>
<td>-2.414 bits/seq.</td>
<td>12.024 bits/seq.</td>
</tr>
<tr>
<td>Bark</td>
<td>-0.55 bits/pixel</td>
<td>1.5 bits/pixel</td>
</tr>
<tr>
<td>Dead Leaves</td>
<td>1.489 bits/pixel</td>
<td>3.536 bits/pixel</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>11.895 bits/pixel</td>
<td>18.037 bits/pixel</td>
</tr>
<tr>
<td>MNIST</td>
<td>See table 2</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. The lower bound $K$ on the log likelihood, computed on a holdout set, for each of the trained models. See Equation 12. The right column is the improvement relative to an isotropic Gaussian or independent binomial distribution. $L_{null}$ is the log likelihood of $\pi (x^{(0)})$.

applies in the case where $r (x^{(t)})$ consists of a delta function for some subset of coordinates, as in the inpainting example in Figure 5.

2.5.4. CHOOSING $r (x^{(t)})$

Typically, $r (x^{(t)})$ should be chosen to change slowly over the course of the trajectory. For the experiments in this paper we chose it to be constant,

$$r (x^{(t)}) = r (x^{(0)}).$$

Another convenient choice is $r (x^{(t)}) = r (x^{(0)}) e^{-t/T}$. Under this second choice $r (x^{(t)})$ makes no contribution to the starting distribution for the reverse trajectory. This guarantees that drawing the initial sample from $\tilde{p} (x^{(T)})$ for the reverse trajectory remains straightforward.

2.6. Entropy of Reverse Process

Since the forward process is known, it is possible to place upper and lower bounds on the entropy of each step in the reverse trajectory. These bounds can be used to constrain the learned reverse transitions $p (x^{(t-1)} | x^{(t)})$. The bounds on the conditional entropy of a step in the reverse trajectory are

$$H_q \left( X^{(t)} | X^{(t-1)} \right) + H_q \left( X^{(t-1)} | X^{(0)} \right) - H_q \left( X^{(t)} | X^{(0)} \right)$$

$$\leq H_q \left( X^{(t-1)} | X^{(t)} \right) \leq H_q \left( X^{(t)} | X^{(t-1)} \right),$$

where both the upper and lower bounds depend only on the conditional forward trajectory $q (x^{(1:T)} | x^{(0)})$, and can be analytically computed. The derivation is provided in Appendix A.

3. Experiments

We train diffusion probabilistic models on a variety of continuous datasets, and a binary dataset. We then demonstrate sampling from the trained model and inpainting of missing data, and compare model performance against other techniques. In all cases the objective function and gradient were computed using Theano (Bergstra & Breuleux, 2010), and model training was with SFO (Sohl-Dickstein et al., 2014). The lower bound on the log likelihood provided by our model is reported for all datasets in Table 1. A reference implementation of the algorithm utilizing Blocks (van Merrienboer et al., 2015) is available at https://github.com/Sohl-Dickstein/Diffusion-Probabilistic-Models.

3.1. Toy Problems

3.1.1. SWISS ROLL

A diffusion probabilistic model was built of a two dimensional swiss roll distribution, using a radial basis function network to generate $f_R (x^{(0)}, t)$ and $f_S (x^{(0)}, t)$. As illustrated in Figure 1, the swiss roll distribution was successfully learned. See Appendix Section D.1.1 for more details.

3.1.2. BINARY HEARTBEAT DISTRIBUTION

A diffusion probabilistic model was trained on simple binary sequences of length 20, where a 1 occurs every 5th time bin, and the remainder of the bins are 0, using a multi-layer perceptron to generate the Bernoulli rates $f_R (x^{(0)}, t)$ of the reverse trajectory. The log likelihood under the true distribution is $\log_2 (\frac{1}{5}) = -2.322$ bits per sequence. As can be seen in Figure 2 and Table 1 learning was nearly perfect. See Appendix Section D.1.2 for more details.

3.2. Images

We trained Gaussian diffusion probabilistic models on several image datasets. The multi-scale convolutional archi-
Deep Unsupervised Learning using Nonequilibrium Thermodynamics

Figure 5. Inpainting. (a) A bark image from (Lazebnik et al., 2005). (b) The same image with the central 100×100 pixel region replaced with isotropic Gaussian noise. This is the initialization $\hat{p}(x^{(T)})$ for the reverse trajectory. (c) The central 100×100 region has been inpainted using a diffusion probabilistic model trained on images of bark, by sampling from the posterior distribution over the missing region conditioned on the rest of the image. Note the long-range spatial structure, for instance in the crack entering on the left side of the inpainted region. The sample from the posterior was generated as described in Section 2.5, where $r(x^{(O)})$ was set to a delta function for known data, and a constant for missing data.

3.2.1. Datasets

**MNIST** In order to allow a direct comparison against previous work on a simple dataset, we trained on MNIST digits (LeCun & Cortes, 1998). The relative log likelihoods are given in Table 2 to a variety of techniques (Bengio et al., 2012; Bengio & Thibodeau-Laufer, 2013; Goodfellow et al., 2014). Samples from the MNIST model are given in Figure App.1 in the Appendix. Our training algorithm provides an asymptotically exact lower bound on the log likelihood. However, most previous reported results on MNIST log likelihood rely on Parzen-window based estimates computed from model samples. For this comparison we therefore estimate MNIST log likelihood using the Parzen-window code released with (Goodfellow et al., 2014).

**CIFAR-10** A probabilistic model was fit to the training images for the CIFAR-10 challenge dataset (Krizhevsky & Hinton, 2009). Samples from the trained model are provided in Figure 3.

**Dead Leaf Images** Dead leaf images (Jeulin, 1997; Lee et al., 2001) consist of layered occluding circles, drawn from a power law distribution over scales. They have an analytically tractable structure, but capture many of the statistical complexities of natural images, and therefore provide a compelling test case for natural image models. As illustrated in Table 2 and Figure 4, we achieve state of the art performance on the dead leaves dataset.

**Bark Texture Images** A probabilistic model was trained on bark texture images (T01-T04) from (Lazebnik et al., 2005). For this dataset we demonstrate that it is straightforward to evaluate or generate from a posterior distribution, by inpainting a large region of missing data using a sample from the model posterior in Figure 5.

4. Conclusion

We have introduced a novel algorithm for modeling probability distributions that enables exact sampling and evaluation of probabilities and demonstrated its effectiveness on a variety of toy and real datasets, including challenging natural image datasets. For each of these tests we used a similar basic algorithm, showing that our method can accurately model a wide variety of distributions. Most existing density estimation techniques must sacrifice modeling power in order to stay tractable and efficient, and sampling or evaluation are often extremely expensive. The core of our algorithm consists of estimating the reversal of a Markov diffusion chain which maps data to a noise distribution; as the number of steps is made large, the reversal distribution of each diffusion step becomes simple and easy to estimate. The result is an algorithm that can learn a fit to any data distribution, but which remains tractable to train, exactly sample from, and evaluate, and under which it is straightforward to manipulate conditional and posterior distributions.

Acknowledgements

We thank Lucas Theis, Subhaneil Lahiri, Ben Poole, Diederik P. Kingma, Taco Cohen, and Philip Bachman for extremely helpful discussion, and Ian Goodfellow for sharing Parzen-window code. We thank Khan Academy and the Office of Naval Research for funding Jascha Sohl-Dickstein. We further thank the
Office of Naval Research, the Burroughs-Wellcome foundation, Sloan foundation, and James S. McDonnell foundation for funding Surya Ganguli.

References


Deep Unsupervised Learning using Nonequilibrium Thermodynamics


