Deep Generative Stochastic Networks Trainable by Backprop

Yoshua Bengio[*] FIND.US@ON.THE.WEB
Éric Thibodeau-Laufer
Guillaume Alain
Département d'informatique et recherche opérationnelle, Université de Montréal,[*] & Canadian Inst. for Advanced Research

Jason Yosinski
Department of Computer Science, Cornell University

Abstract

We introduce a novel training principle for probabilistic models that is an alternative to maximum likelihood. The proposed Generative Stochastic Networks (GSN) framework is based on learning the transition operator of a Markov chain whose stationary distribution estimates the data distribution. The transition distribution of the Markov chain is conditional on the previous state, generally involving a small move, so this conditional distribution has fewer dominant modes, being unimodal in the limit of small moves. Thus, it is easier to learn because it is easier to approximate its partition function, more like learning to perform supervised function approximation, with gradients that can be obtained by backprop. We provide theorems that generalize recent work on the probabilistic interpretation of denoising autoencoders and obtain along the way an interesting justification for dependency networks and generalized pseudolikelihood, along with a definition of an appropriate joint distribution and sampling mechanism even when the conditionals are not consistent. GSNs can be used with missing inputs and can be used to sample subsets of variables given the rest. We validate these theoretical results with experiments on two image datasets using an architecture that mimics the Deep Boltzmann Machine Gibbs sampler but allows training to proceed with simple backprop, without the need for layerwise pretraining.

1. Introduction

Research in deep learning (see Bengio (2009) and Bengio et al. (2013a) for reviews) grew from breakthroughs in unsupervised learning of representations, based mostly on the RBM (Hinton et al., 2006), AE variants (Bengio et al., 2007;

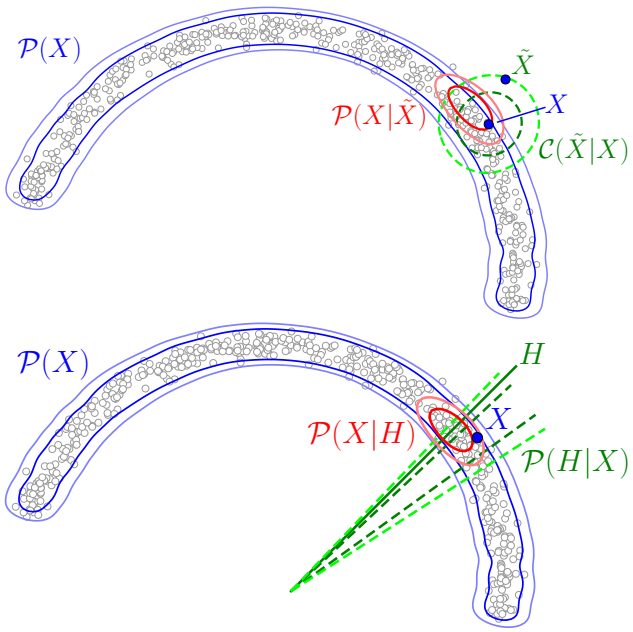


Figure 1. Top: A denoising AE defines an estimated Markov chain where the transition operator first samples a corrupted $X^{\tilde{}}$ from $\mathcal{C}(X^{\tilde{}}|X)$ and then samples a reconstruction from $P_\theta(X|X^{\tilde{}})$, which is trained to estimate the ground truth $P(X|X^{\tilde{}})$. Note how for any given $X^{\tilde{}}$, $P(X|X^{\tilde{}})$ is a much simpler (roughly unimodal) distribution than the ground truth $P(X)$ and its partition function is thus easier to approximate . *Bottom:* More generally , a GSN allows the use of arbitrary LVs H in addition to X, with the Markov chain state (and mixing) involving both X and H. Here H is the angle about the origin. The GSN inherits the benefit of a simpler conditional and adds LVs , which allow far more powerful deep representations in which mixing is easier (Bengio et al., 2013b).

Proceedings of the 31[st] International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

Vincent et al., 2008), and sparse coding variants (Lee et al., 2007; Ranzato et al., 2007). However, the most impressive recent results have been obtained with purely supervised learning techniques for deep networks, in particular for speech recognition (Dahl et al., 2010; Deng et al., 2010; Seide et al., 2011) and object recognition (Krizhevsky et al., 2012). The latest breakthrough in object recognition (Krizhevsky et al., 2012) was achieved with fairly deep convolutional networks with a form of noise injection in the input and hidden layers during training, called dropout (Hinton et al., 2012).

Though the goal of training large unsupervised networks has turned out to be more elusive than its supervised counterpart, the vastly larger available volume of unlabeled data still beckons for efficient methods to model it. Recent progress in training supervised models raises the question: can we take advantage of this progress to improve our ability to train deep, generative, unsupervised, semi-supervised or structured output models?

This paper lays theoretical foundations for a move in this direction through the following main contributions:

1 – Intuition: In Section 2 we discuss what we view as basic motivation for studying alternate ways of training unsupervised probabilistic models, i.e., avoiding the intractable sums or maximization involved in many approaches.

2 – Training Framework: We generalize recent work on the generative view of denoising AEs (Bengio et al., 2013c) by introducing LVs in the framework to define GSNs (Section 3). GSNs aim to estimate the data generating distribution indirectly, by parametrizing the transition operator of a Markov chain rather than directly parametrizing $P(X)$. Most critically, *this framework transforms the unsupervised density estimation problem into one which is more similar to supervised function approximation*. This enables training by (possibly regularized) maximum likelihood and gradient descent computed via simple back-propagation, avoiding the need to compute intractable partition functions. Depending on the model, this may allow us to draw from any number of recently demonstrated supervised training tricks.

3 – General theory: Training the generative (decoding / denoising) component of a GSN $P(X|h)$ with noisy representation h is often far easier than modeling $P(X)$ explicitly (compare the blue and red distributions in Figure 1). We prove that if our estimated $P(X|h)$ is consistent (e.g. through maximum likelihood), then the stationary distribution of the resulting chain is a consistent estimator of the data generating density, $P(X)$ (Section 3.2). We strengthen the consistency theorems introduced in Bengio et al. (2013c) by showing that the corruption distribution may be purely local, not requiring support over the whole domain of the visible variables (Section 3.1).

4 – Consequences of theory: We show that the model is general and extends to a wide range of architectures, including sampling procedures whose computation can be unrolled as a Markov Chain, i.e., architectures that add noise during intermediate computation in order to produce random samples of a desired distribution (Theorem 2). An exciting frontier in machine learning is the problem of modeling so-called structured outputs, i.e., modeling a conditional distribution where the output is high-dimensional and has a complex multimodal joint distribution (given the input variable). We show how GSNs can be used to support such structured output and missing values (Section 3.3).

5 – Example application: In Section 4 we show an example application of the GSN theory to create a deep GSN whose computational graph resembles the one followed by Gibbs sampling in deep Bms (with continuous LVs), but that can be trained efficiently with back-propagated gradients and without layerwise pretraining. Because the Markov Chain is defined over a state (X, h) that includes LVs, we reap the dual advantage of more powerful models for a given number of parameters and better mixing in the chain as we add noise to variables representing higher-level information, first suggested by the results obtained by Bengio et al. (2013b) and Luo et al. (2013). The experimental results show that such a model with latent states indeed mixes better than shallower models without them (Table 1).

6 – Dependency networks: Finally, an unexpected result falls out of the GSN theory: it allows us to provide a novel justification for dependency networks (Heckerman et al., 2000) and for the first time define a proper joint distribution between all the visible variables that is learned by such models (Section 3.4).

2. Summing over too many major modes

Many of the computations involved in graphical models (inference, sampling, and learning) are made intractable and difficult to approximate because of the large number of non-negligible modes in the modeled distribution (either directly $P(x)$ or a joint distribution $P(x, h)$ involving latent variables h). In all of these cases, what is intractable is the computation or approximation of a sum (often weighted by probabilities), such as a marginalization or the estimation of the gradient of the normalization constant. If only a few terms in this sum dominate (corresponding to the dominant modes of the distribution), then many good approximate methods can be found, such as MCMC methods.

Similarly difficult tasks arise with structured output problems where one wants to sample from $P(y, h|x)$ and both y and h are high-dimensional and have a complex highly multimodal joint distribution (given x).

Deep BMs (Salakhutdinov & Hinton, 2009) combine the difficulty of inference (for the *positive phase* where one tries to push the energies associated with the observed x down) and also that of sampling (for the *negative phase* where one tries to push up the energies associated with x's sampled from $P(x)$). Unfortunately, using an MCMC method to sample from $P(x, h)$ in order to estimate the gradient of the partition function may be seriously hurt by the presence of a large number of

important modes, as argued below.

To evade the problem of highly multimodal joint or posterior distributions, the currently known approaches to dealing with the above intractable sums make very strong explicit assumptions (in the parametrization) or implicit assumptions (by the choice of approximation methods) on the form of the distribution of interest. In particular, MCMC methods are more likely to produce a good estimator if the number of non-negligible modes is small: otherwise the chains would require at least as many MCMC steps as the number of such important modes, times a factor that accounts for the mixing time between modes. Mixing time itself can be very problematic as a trained model becomes sharper, as it approaches a data generating distribution that may have well-separated and sharp modes (i.e., manifolds).

We propose to make another assumption that might suffice to bypass this multimodality problem: the effectiveness of function approximation.

In particular, the GSN approach presented in the next section relies on estimating the transition operator of a Markov chain, e.g. $P(x_t|x_{t-1})$ or $P(x_t, h_t|x_{t-1}, h_{t-1})$. Because each step of the Markov chain is generally local, these transition distributions will often include only a very small number of important modes (those in the neighbourhood of the previous state). Hence the gradient of their partition function will be easy to approximate. For example consider the denoising transitions studied by Bengio et al. (2013c) and illustrated in Figure 1, where $\tilde{x}_{t-1}$ is a stochastically corrupted version of x_{t-1} and we learn the denoising distribution $P(x|\tilde{x})$. In the extreme case (studied empirically here) where $P(x|\tilde{x})$ is approximated by a unimodal distribution, the only form of training that is required involves function approximation (predicting the clean x from the corrupted $\tilde{x}$).

Although having the true $P(x|\tilde{x})$ turn out to be unimodal makes it easier to find an appropriate family of models for it, unimodality is by no means required by the GSN framework itself. One may construct a GSN using any multimodal model for output (e.g. MoGs, RBMs, NADE, etc .), provided that gradients for the parameters of the model in question can be estimated (e.g. log-likelihood gradients).

The approach proposed here thus avoids the need for a poor approximation of the gradient of the partition function in the inner loop of training, but still has the potential of capturing very rich distributions by relying mostly on "function approximation".

Besides the approach discussed here, there may well be other very different ways of evading this problem of intractable marginalization, including approaches such as sum-product networks (Poon & Domingos, 2011), which are based on learning a probability function that has a tractable form by construction and yet is from a flexible enough family of distributions.

3. GSNs

Assume the problem we face is to construct a model for some unknown data-generating distribution $P(X)$ given only examples of X drawn from that distribution. In many cases, the unknown distribution $P(X)$ is complicated, and modeling it directly can be difficult.

A recently proposed approach using denoising AEs transforms the difficult task of modeling $P(X)$ into a supervised learning problem that may be much easier to solve. The basic approach is: given a clean example data point $X \sim P(X)$, we obtain a corrupted version $X^{\sim}$ by sampling from some corruption distribution $\mathcal{C}(X^{\sim}|X)$. For example, we might take a clean image, X, and add random white noise to produce $X^{\sim}$. We then use supervised learning methods to train a function to reconstruct, as accurately as possible, any X from the data set given only a noisy version $X^{\sim}$. As shown in Figure 1, the reconstruction distribution $P(X|X^{\sim})$ may often be much easier to learn than the data distribution $P(X)$, *because* $P(X|X^{\sim})$ *tends to be dominated by a single or few major modes* (such as the roughly Gaussian shaped density in the figure).

But how does learning the reconstruction distribution help us solve our original problem of modeling $P(X)$? The two problems are clearly related, because if we knew everything about $P(X)$, then our knowledge of the $\mathcal{C}(\tilde{X}|X)$ that we chose would allow us to precisely specify the optimal reconstruction function via Bayes rule: $P(X|\tilde{X}) = \frac{1}{z}\mathcal{C}(\tilde{X}|X)P(X)$, where z is a normalizing constant that does not depend on X. As one might hope, the relation is also true in the opposite direction: once we pick a method of adding noise, $\mathcal{C}(\tilde{X}|X)$, knowledge of the corresponding reconstruction distribution $P(X|\tilde{X})$ is sufficient to recover the density of the data $P(X)$.

This intuition was borne out by proofs in two recent papers. Alain & Bengio (2013) showed that denoising AEs with small Gaussian corruption and squared error loss estimated the score (derivative of the log-density wrt the input) of continuous observed random variables. More recently, Bengio et al. (2013c) generalized this to arbitrary variables (discrete, continuous or both), arbitrary corruption (not necessarily asymptotically small), and arbitrary loss function (so long as they can be seen as a log-likelihood).

Beyond proving that $P(X|\tilde{X})$ is sufficient to reconstruct the data density, Bengio et al. (2013c) also demonstrated a method of sampling from a learned, parametrized model of the density, $P_\theta(X)$, by running a Markov chain that alternately adds noise using $\mathcal{C}(\tilde{X}|X)$ and denoises by sampling from the learned $P_\theta(X|\tilde{X})$, which is trained to approximate the true $P(X|\tilde{X})$. The most important contribution of that paper was demonstrating that if a learned, parametrized reconstruction function $P_\theta(X|\tilde{X})$ converges to the true $P(X|\tilde{X})$, then under some relatively benign conditions the stationary distribution $\pi(X)$ of the resulting Markov chain will exist and will indeed converge

to the data distribution $P(X)$.

Before moving on, we should pause to make an important point clear. Alert readers may have noticed that $P(X|\tilde{X})$ and $P(X)$ can each be used to reconstruct the other given knowledge of $\mathcal{C}(\tilde{X}|X)$. Further, if we assume that we have chosen a simple $\mathcal{C}(\tilde{X}|X)$ (say, a uniform Gaussian with a single width parameter), then $P(X|\tilde{X})$ and $P(X)$ must both be of approximately the same complexity. Put another way, we can never hope to combine a simple $\mathcal{C}(\tilde{X}|X)$ and a simple $P(X|\tilde{X})$ to model a complex $P(X)$. Nonetheless, it may still be the case that $P(X|\tilde{X})$ is easier to *model* than $P(X)$ due to reduced computational complexity in computing or approximating the partition functions of the conditional distribution mapping corrupted input $\tilde{X}$ to the distribution of corresponding clean input X. Indeed, because that conditional is going to be mostly assigning probability to X locally around $\tilde{X}$, $P(X|\tilde{X})$ has only one or a few modes, while $P(X)$ can have a very large number.

So where did the complexity go? $P(X|\tilde{X})$ has fewer modes than $P(X)$, but *the location of these modes depends on the value of* $\tilde{X}$. It is precisely this mapping from $\tilde{X} \rightarrow$ *mode location* that allows us to trade a difficult density modeling problem for a supervised function approximation problem that admits application of many of the usual supervised learning tricks.

In the next four sections, we extend previous results in several directions.

3.1. Generative denoising AEs with local noise

The main theorem in Bengio et al. (2013c) (stated in Theorem S1) requires that the Markov chain be ergodic. A set of conditions guaranteeing ergodicity is given in the aforementioned paper, but these conditions are restrictive in requiring that $\mathcal{C}(\tilde{X}|X) > 0$ everywhere that $P(X) > 0$. The effect of these restrictions is that $P_\theta(X|\tilde{X})$ must have the capacity to model every mode of $P(X)$, exactly the difficulty we were trying to avoid. We show in this paper's supplemental material how we may also achieve the required ergodicity through other means, allowing us to choose a $\mathcal{C}(\tilde{X}|X)$ that only makes small jumps, which in turn only requires $P_\theta(X|\tilde{X})$ to model a small part of the space around each $\tilde{X}$.

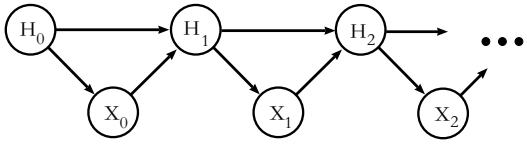
3.2. Generalizing the denoising AE to GSNs

The denoising AE Markov chain is defined by

$\tilde{X}_t \sim C(\tilde{X}|X_t)$ and $X_{t+1} \sim P_\theta(X|\tilde{X}_t)$, where X_t alone can serve as the state of the chain.

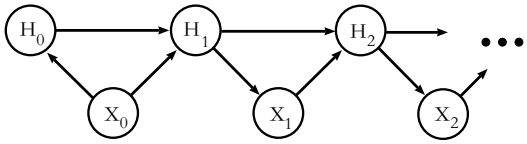
The GSN framework generalizes this by defining a Markov chain with both a visible X_t and a LV H_t as state variables, of the form

$$H_{t+1} \sim P_{\theta_1}(H|H_t, X_t)$$
$$X_{t+1} \sim P_{\theta_2}(X|H_{t+1}).$$



Denoising AEs are thus a special case of GSNs. Note that, given that the distribution of H_{t+1} depends on a previous value of H_t, we find ourselves with an extra H_0 variable added at the beginning of the chain. This H_0 complicates things when it comes to training, but when we are in a sampling regime we can simply wait a sufficient number of steps to burn in. The next theoretical results give conditions for making the stationary distributions of the above Markov chain match a target data generating distribution.

Theorem 2. *Let $(H_t, X_t)_{t=0}^{\infty}$ be the Markov chain defined by the following graphical model.*



the chain has a stationary distribution $\pi_{H,X}$, and

- *all the $P(X_t = x|H_t = h) = g(x, h)$ share the same density for $t \geq 1$*
- *all the $P(H_{t+1} = h|H_t = h', X_t = x) = f(h, h', x)$ share the same density for $t \geq 0$*
- *$P(H_0 = h|X_0 = x) = P(H_1 = h|X_0 = x)$*
- *$P(X_1 = x|H_1 = h) = P(X_0 = x|H_1 = h)$*

=> P(X0, H0) = P(X1, H1)

==>

- *$P(X_0{=}x|H_0{=}h){=}g(x, h)$,*
- *$P(X_t = x, H_t = h) = P(X_0 = x, H_0 = h)$ for all $t \geq 0$*
- *the stationary distribution $\pi_{H,X}$ has a marginal distribution π_X such that $\pi(x) = P(X_0 = x)$.*

Those conclusions show that our Markov chain has the property that its samples in X are drawn from the same distribution as X_0.

We also address there the issue of the consistency of the stationary distribution that we obtain when using an increasing, but finite, number of training samples.

We avoid discussing the training criterion for a GSN. Various alternatives exist, but this analysis is for future work. Right now Theorem 2 suggests the following rules :

- Pick the transition distribution $f(h, h', x)$ to be useful (e.g. through training that maximizes reconstruction likelihood).
- Make sure that during training $P(H_0{=}h|X_0{=}x){=}P(H_1 = h|X_0 = x)$. One interesting way to achieve this is, for each X_0 in the training set, iteratively sample

$H_1|(H_0, X_0)$ and substitute the value of H_1 as the updated value of H_0. Repeat until you have achieved a kind of "burn in". Note that, after the training is completed, when we use the chain for sampling, the samples that we get from its stationary distribution do not depend on H_0. This technique of substituting the H_1 into H_0 does not apply beyond the training step.

- Define $g(x, h)$ to be your estimator for $P(X_0 = x|H_1 = h)$, e.g. by training an estimator of this conditional distribution from the samples (X_0, H_1).
- The rest of the chain for $t \geq 1$ is defined in terms of (f, g).

There several equivalent ways of expressing a GSN. One of the interesting formulations is to use deterministic functions of random variables to express the densities (f, g) used in Theorem 2. With that approach, we define $H_{t+1} = f_{\theta_1}(X_t, Z_t, H_t)$ for some independent noise source Z_t, and we insist that X_t cannot be recovered exactly from H_{t+1}. The advantage of that formulation is that one can directly back-propagated the reconstruction log-likelihood $\log P(X_1 = x_0|H_1 = f(X_0, Z_0, H_0))$ into all the parameters of f and g (a similar idea was independently proposed in (Kingma, 2013) and also exploited in (Rezende et al., 2014)).

For the rest of this paper, we will use such a deterministic function f instead of having f refer to a PDF.

In the setting described at the beginning of section 3, the function playing the role of the "encoder" was fixed for the purpose of the theorem, and we showed that learning only the "decoder" part (but a sufficiently expressive one) sufficed. In this setting we are learning both, which can cause certain broken behavior.

One problem would be if the created Markov chain failed to converge to a stationary distribution. Another such problem could be that the function $f(X_t, Z_t, H_t)$ learned would try to ignore the noise Z_t, or not make the best use out of it. In that case, the reconstruction distribution would simply converge to a Dirac at the input X. This is the analogue of the constraint on AEs that is needed to prevent them from learning the identity function . Here , we must design the family from which f and g are learned such that when the noise Z is injected , there are always several possible values of X that could have been the correct original input.

Another extreme case to think about is when $f(X, Z, H)$ is overwhelmed by the noise and has lost all information about X. In that case the theorems are still applicable while giving uninteresting results: the learner must capture the full distribution of X in $P_{\theta_2}(X|H)$ because the latter is now equivalent to $P_{\theta_2}(X)$, since $f(X, Z, H)$ no longer contains information about X. This illustrates that when the noise is large, the reconstruction distribution (parametrized by θ_2) will need to have the expressive power to represent multiple modes. Otherwise, the reconstruction will tend to capture an average output, which would visually look like a fuzzy combination of actual modes. In the experiments performed here, we have only considered unimodal reconstruction distributions (with factorized outputs), because we expect that even if $P(X|H)$ is not unimodal, it would be dominated by a single mode when the noise level is small. However, future work should investigate multimodal alternatives.

A related element to keep in mind is that one should pick the family of conditional distributions $P_{\theta_2}(X|H)$ so that one can sample from them and one can easily train them when given (X, H) pairs, e.g., by maximum likelihood.

3.3. Handling missing inputs or structured output

In general, a simple way to deal with missing inputs is to clamp the observed inputs and then apply the Markov chain with the constraint that the observed inputs are fixed and not resampled at each time step, whereas the unobserved inputs are resampled each time, *conditioned on the clamped inputs*.

Proposition 1. **If** *a subset $x^{(s)}$ of the elements of X is kept fixed (not resampled) while the remainder $X^{(-s)}$ is updated stochastically during the Markov chain of Theorem 2, but using $P(X_t|H_t, X_t^{(s)} = x^{(s)})$,* **then** *the asymptotic distribution π_n of the Markov chain produces samples of $X^{(-s)}$ from the conditional distribution $\pi_n(X^{(-s)}|X^{(s)} = x^{(s)})$.*

Practically, it means that we must choose an output (reconstruction) distribution from which it is not only easy to sample from, but also from which it is easy to sample a subset of the variables in the vector X *conditioned on the rest being known*. In the experiments below, we used a factorial distribution for the reconstruction, from which it is trivial to sample conditionally a subset of the input variables. In general (with non-factorial output distributions) one must use the proper conditional for the theorem to apply, i.e., it is not sufficient to clamp the inputs, one must also sample the reconstructions from the appropriate conditional distribution (conditioning on the clamped values).

This method of dealing with missing inputs can be immediately applied to structured outputs. If $X^{(s)}$ is viewed as an "input" and $X^{(-s)}$ as an "output", then sampling from $X_{t+1}^{(-s)} \sim P(X^{(-s)}|f((X^{(s)}, X_t^{(-s)}), Z_t, H_t), X^{(s)})$ will converge to estimators of $P(X^{(-s)}|X^{(s)})$. This still requires good choices of the parametrization (for f as well as for the conditional probability P), but the advantages of this approach are that there is no approximate inference of LVs and the learner is trained wrt simpler conditional probabilities : in the limit of small noise , we conjecture that these conditional probabilities can be well approximated by unimodal distributions . Theoretical evidence comes from Alain & Bengio (2013): *when the amount of corruption noise converges to 0 and the input variables have a smooth continuous density, then a unimodal Gaussian reconstruction density suffices to fully capture the joint distribution.*

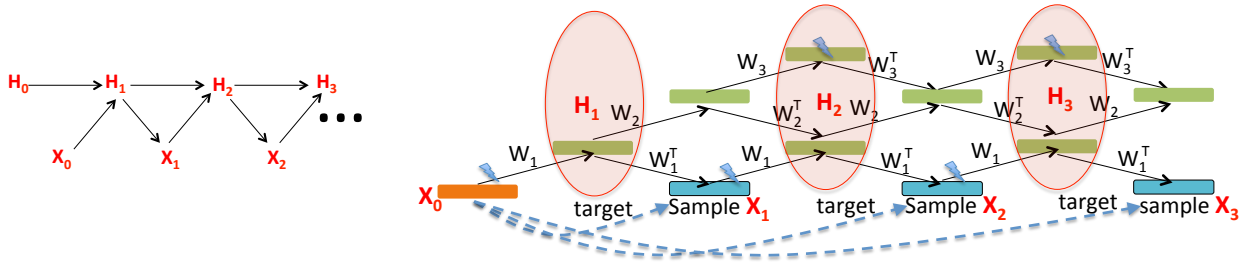


Figure 2. Left: Generic GSN Markov chain with state variables X_t and H_t. *Right:* GSN Markov chain inspired by the unfolded computational graph of the DBM Gibbs sampling process, but with back-propable stochastic units at each layer. The training example $X = x_0$ starts the chain. Either odd or even layers are stochastically updated at each step. All x_t's are corrupted by salt-and-pepper noise before entering the graph (lightning symbol). Each x_t for $t > 0$ is obtained by sampling from the reconstruction distribution for that step, $P_{\theta_2}(X_t|H_t)$. The walkback training objective is the sum over all steps of log-likelihoods of target $X = x_0$ under the reconstruction distribution. In the special case of a unimodal Gaussian reconstruction distribution, maximizing the likelihood is equivalent to minimizing reconstruction error; in general one trains to maximum likelihood, not simply minimum reconstruction error.

3.4. Dependency Networks as GSNs

Dependency networks (Heckerman et al., 2000) are models in which one estimates conditionals $P_i(x_i|x_{-i})$, where x_{-i} denotes $x \setminus x_i$, i.e., the set of variables other than the i-th one, x_i. Note that each P_i may be parametrized separately, thus not guaranteeing that there exists a joint of which they are the conditionals. Instead of the ordered pseudo-Gibbs sampler defined in Heckerman et al. (2000), which resamples each variable x_i in the order $x_1, x_2, \ldots$, we can view dependency networks in the GSN framework by defining a proper Markov chain in which at each step one randomly chooses which variable to resample. The corruption process therefore just consists of $H = f(X, Z) = X_{-s}$ where X_{-s} is the complement of X_s, with s a randomly chosen subset of elements of X (possibly constrained to be of size 1). Furthermore, we parametrize the reconstruction distribution as $P_{\theta_2}(X = x|H) = \delta_{x_{-s}=X_{-s}} P_{\theta_2,s}(X_s = x_s|x_{-s})$ where the estimated conditionals $P_{\theta_2,s}(X_s = x_s|x_{-s})$ are not constrained to be consistent conditionals of some joint distribution over all of X.

Proposition 2. *If the above GSN Markov chain has a stationary distribution, then the dependency network defines a joint distribution (which is that stationary distribution), which does not have to be known in closed form. Furthermore, if the conditionals are consistent estimators of the ground truth conditionals, then that stationary distribution is a consistent estimator of the ground truth joint distribution.*

The proposition can be proven by immediate application of Theorem 1 from Bengio et al. (2013c) with the above definitions of the GSN. This joint stationary distribution can exist even if the conditionals are not consistent. To show that, assume that some choice of (possibly inconsistent) conditionals gives rise to a stationary distribution π. Now let us consider the set of all conditionals (not necessarily consistent) that could have given rise to that π. Clearly, the conditionals derived from π is part of that set, but there are infinitely many others (a simple counting argument shows that the fixed point equation of π introduces fewer constraints than the number of degrees of freedom that define the conditionals). To better understand why the ordered pseudo-Gibbs chain does not benefit from the same properties, we can consider an extended case by adding an extra component of the state X, being the index of the next variable to resample. In that case, the Markov chain associated with the ordered pseudo-Gibbs procedure would be periodic, thus violating the ergodicity assumption of the theorem. However, by introducing randomness in the choice of which variable(s) to resample next, we obtain aperiodicity and ergodicity, yielding as stationary distribution a mixture over all possible resampling orders. These results also show in a novel way (see e.g. Hyvärinen (2006) for earlier results) that training by pseudolikelihood or generalized pseudolikelihood provides a consistent estimator of the associated joint, so long as the GSN Markov chain defined above is ergodic. This result can be applied to show that the multi-prediction deep Boltzmann machine (MP-DBM) training procedure introduced by Goodfellow et al. (2013) also corresponds to a GSN. This has been exploited in order to obtain much better samples using the associated GSN Markov chain than by sampling from the corresponding DBM (Goodfellow et al., 2013). Another interesting conclusion that one can draw from this paper and its GSN interpretation is that sota classification error can thereby be obtained: 0.91% on MNIST without fine-tuning (best comparable previous DBM results was well above 1%) and 10.6% on permutation-invariant NORB (best previous DBM results was 10.8%).

4. Experimental Example of GSN

The theoretical results on GSNs open for exploration a large class of possible parametrizations which will share the property that they can capture the underlying data distribution through the GSN Markov chain. What parametrizations will work well? Where and how should one inject noise? We present results of preliminary experiments with specific selections for each of these choices, but the reader should keep in mind that the space of possibilities is vast.

As a conservative starting point, we propose to explore families of parametrizations which are similar to existing deep stochastic architectures such as the DBM(Salakhutdinov & Hinton , 2009). Basically , the idea is to construct a computational graph that is similar to the computational graph for Gibbs sampling or variational inference in DBMs . However , we have to diverge a bit from these architectures in order to accommodate the desirable property that it will be possible to back-propagate the gradient of reconstruction log-likelihood wrt the parameters θ_1 and θ_2. Since the gradient of a binary stochastic nunit is 0 almost everywhere , we have to consider related alternatives . An interesting source of inspiration regarding this question is a recent paper on estimating or propagating gradients through stochastic neurons (Bengio , 2013). Here we consider the following stochastic non-linearities : $h_i = \eta_{\text{out}} + \tanh(\eta_{\text{in}} + a_i)$ where a_i is the linear activation for unit i (an affine transformation applied to the input of the unit , coming from the layer below , the layer above , or both) and η_{in} and η_{out} are zero-mean Gaussian noises.

To emulate a sampling procedure similar to BMs in which the filled-in missing values can depend on the representations at the top level , the computational graph allows information to propagate both upwards (from input to higher levels) and downwards , giving rise to the computational graph structure illustrated in Figure 2, which is similar to that explored for *deterministic* recurrent AEs (Seung , 1998 ; Behnke , 2001 ; Savard, 2011). Downward weight matrices have been fixed to the transpose of corresponding upward weight matrices.

The *walkback* algorithm was proposed in Bengio et al. (2013 c) to make training of generalized denoising AEs (a special case of the models studied here) more efficient . The basic idea is that the reconstruction is obtained after not one but several steps of the sampling Markov chain. In this context it simply means that the computational graph from X to a reconstruction probability actually involves generating intermediate samples as if we were running the Markov chain starting at X . In the experiments , the graph was unfolded so that $2D$ sampled reconstructions would be produced , where D is the depth (number of hidden layers). The training loss is the sum of the reconstruction negative log-likelihoods (of target X) over all those reconstruction steps.

The supplemental material provides full details on the experiments and more detailed figures of generated samples. We summarize the results here. The experiments were performed on the MNIST and Toronto Face Database (TFD) (Susskind et al., 2010) datasets, following the setup in Bengio et al. (2013b), where the model generates quantized (binary) pixels. A lower bound on the log-likelihood, based only on the generated samples (or rather the conditional expectations $E[X|H]$ for the sampled H's) and introduced in Breuleux et al. (2011) was used to compare various models in Table 1. The results show that a GSN with latent state performed better than a pure denoising AE (or equivalently, that a deeper GSN yields both better samples and a better likelihood bound). As can be seen, the samples are of quality comparable to those obtained by DBMs and Deep Belief Nets (DBNs). Figures 3 and 4 illustrate generated samples and show the fast mixing . Figure 3 (bottom) also shows successful conditional sampling of the lhs of the image given the rhs.

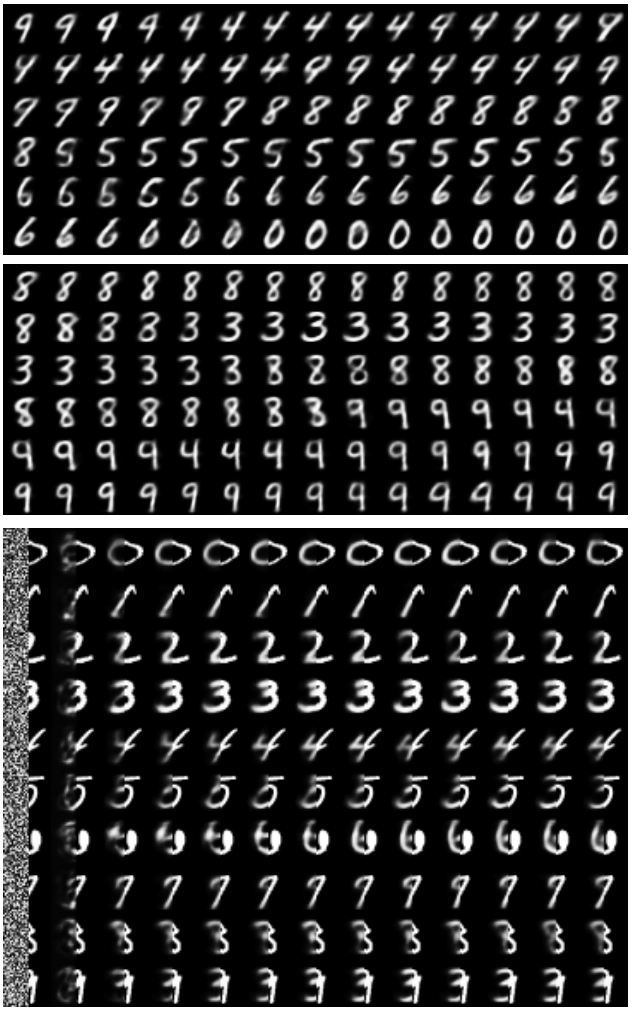


Figure 3. Top: two runs of consecutive samples (one row after the other) generated from 2-layer GSN model, showing fast mixing between classes, nice and sharp images. Note: only every fourth sample is shown; see the supplemental material for the samples in between. Bottom: conditional Markov chain, with the right half of the image clamped to one of the MNIST digit images and the left half successively resampled, illustrating the power of the generative model to stochastically fill-in missing inputs.

5. Conclusion

We have introduced a new approach to training generative models , GSN , that is an alternative to maximum likelihood , with the objective of avoiding the intractable marginalizations and the danger of poor approximations of these marginalizations.

Figure 4. GSN samples from a 3-layer model trained on the TFD dataset. Every second sample is shown; see supplemental material for every sample. At the end of each row, we show the nearest example from the training set to the last sample on that row, to illustrate that the distribution is not merely copying the training set.

Table 1. Test set log-likelihood lower bound (LL) obtained by a Parzen density estimator constructed using 10000 generated samples, for different generative models trained on MNIST. The LL is not directly comparable to AIS likelihood estimates because we use a Gaussian mixture rather than a Bernoulli mixture to compute the likelihood, but we can compare with Rifai et al. (2012); Bengio et al. (2013b;c) (from which we took the last three columns). A DBN-2 has 2 hidden layers, a CAE-1 has 1 hidden layer, and a CAE-2 has 2. The DAE is basically a GSN-1, with no injection of noise inside the network.

	GSN-2	DAE	DBN-2	CAE-1	CAE-2
LL	214	144	138	68	121
STD.ERR.	1.1	1.6	2.0	2.9	1.6

The training procedure is more similar to function approximation than to unsupervised learning because the reconstruction distribution is simpler than the data distribution, often unimodal (provably so in the limit of very small noise). This makes it possible to train unsupervised models that capture the data-generating distribution simply using backprop and gradient descent (in a computational graph that includes noise injection). The proposed theoretical results state that under mild conditions (in particular that the noise injected in the networks prevents perfect reconstruction), training the model to denoise and reconstruct its observations (through a powerful family of reconstruction distributions) suffices to capture the data-generating distribution through a simple Markov chain. Another way to put it is that we are training the transition operator of a Markov chain whose stationary distribution estimates the data distribution, and it turns out that this is a much easier learning problem because the normalization constant for this conditional distribution is generally dominated by fewer modes. These theoretical results are extended to the case where the corruption is local but still allows the chain to mix and to the case where some inputs are missing or constrained (thus allowing to sample from a conditional distribution on a subset of the observed variables or to learned structured output models). The GSN framework is shown to lend to dependency networks a valid estimator of the joint distribution of the observed variables even when the learned conditionals are not consistent, also allowing to prove consistency of generalized pseudolikelihood training, associated with the stationary distribution of the corresponding GSN (that randomly chooses a subset of variables and then resamples it). Experiments have been conducted to validate the theory, in the case where the GSN architecture emulates the Gibbs sampling process of a DBM, on two datasets. A quantitative evaluation of the samples confirms that the training procedure works very well (in this case allowing us to train a deep generative model without layerwise pretraining) and can be used to perform conditional sampling of a subset of variables given the rest.

Acknowledgements

Vincent Dumoulin, Pascal Vincent, Yao Li, Aaron Courville, Ian Goodfellow, and Hod Lipson, as well as funding from NSERC, CIFAR (YB is a CIFAR Senior Fellow), NASA (JY is a Space Technology Research Fellow), and the Canada Research Chairs.

References

Alain, Guillaume and Bengio, Yoshua. What regularized auto-encoders learn from the data generating distribution. In *International Conference on Learning Representations (ICLR'2013)*, 2013.

Behnke, Sven. Learning iterative image reconstruction in the neural abstraction pyramid. *Int. J. Computational Intelligence and Applications*, 1(4):427–438, 2001.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. Greedy layer-wise training of deep networks. In *NIPS'2006*, 2007.

Bengio, Yoshua. *Learning deep architectures for AI*. Now Publishers, 2009.

Bengio, Yoshua. Estimating or propagating gradients through stochastic neurons. Technical Report arXiv:1305.2982, Universite de Montreal, 2013.

Bengio, Yoshua, Courville, Aaron, and Vincent, Pascal. Unsupervised feature learning and deep learning: A review and new perspectives. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 2013a.

Bengio, Yoshua, Mesnil, Grégoire, Dauphin, Yann, and Rifai, Salah. Better mixing via deep representations. In *ICML'13*, 2013b.

Bengio, Yoshua, Yao, Li, Alain, Guillaume, and Vincent, Pascal. Generalized denoising auto-encoders as generative models. In *NIPS26*. Nips Foundation, 2013c.

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

Dahl, George E., Ranzato, Marc'Aurelio, Mohamed, Abdel-rahman, and Hinton, Geoffrey E. Phone recognition with the mean-covariance restricted Boltzmann machine. In *NIPS'2010*, 2010.

Deng, L., Seltzer, M., Yu, D., Acero, A., Mohamed, A., and Hinton, G. Binary coding of speech spectrograms using a deep auto-encoder. In *Interspeech 2010*, Makuhari, Chiba, Japan, 2010.

Goodfellow, Ian J., Mirza, Mehdi, Courville, Aaron, and Bengio, Yoshua. Multi-prediction deep Boltzmann machines. In *NIPS26*. Nips Foundation, 2013.

Heckerman, David, Chickering, David Maxwell, Meek, Christopher, Rounthwaite, Robert, and Kadie, Carl. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1:49–75, 2000.

Hinton, Geoffrey E., Osindero, Simon, and Teh, Yee Whye. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Hinton, Geoffrey E., Srivastava, Nitish, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Improving neural networks by preventing co-adaptation of feature detectors. Technical report, arXiv:1207.0580, 2012.

Hyvärinen, Aapo. Consistency of pseudolikelihood estimation of fully visible boltzmann machines. *Neural Computation*, 2006.

Kingma, Diederik P. Fast gradient-based inference with continuous latent variable models in auxiliary form. Technical report, arXiv:1306.0733, 2013.

Krizhevsky, A., Sutskever, I., and Hinton, G. ImageNet classification with deep convolutional neural networks. In *NIPS'2012*. 2012.

Lee, Honglak, Battle, Alexis, Raina, Rajat, and Ng, Andrew. Efficient sparse coding algorithms. In *NIPS'06*, pp. 801–808. MIT Press, 2007.

Luo, Heng, Carrier, Pierre Luc, Courville, Aaron, and Bengio, Yoshua. Texture modeling with convolutional spike-and-slab RBMs and deep extensions. In *AISTATS'2013*, 2013.

Poon, Hoifung and Domingos, Pedro. Sum-product networks: A new deep architecture. In *UAI'2011*, Barcelona, Spain, 2011.

Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. Efficient learning of sparse representations with an energy-based model. In *NIPS'2006*, 2007.

Rezende, Danilo J., Mohamed, Shakir, and Wierstra, Daan. Stochastic backpropagation and approximate inference in deep generative models. Technical report, arXiv:1401.4082, 2014.

Rifai, Salah, Bengio, Yoshua, Dauphin, Yann, and Vincent, Pascal. A generative process for sampling contractive auto-encoders. In *ICML'12*, 2012.

Salakhutdinov, Ruslan and Hinton, Geoffrey E. Deep Boltzmann machines. In *AISTATS'2009*, pp. 448–455, 2009.

Savard, François. Réseaux de neurones à relaxation entraînés par critère d'autoencodeur débruitant. Master's thesis, U. Montréal, 2011.

Seide, Frank, Li, Gang, and Yu, Dong. Conversational speech transcription using context-dependent deep neural networks. In *Interspeech 2011*, pp. 437–440, 2011.

Seung, Sebastian H. Learning continuous attractors in recurrent networks. In *NIPS'97*, pp. 654–660. MIT Press, 1998.

Susskind, Joshua, Anderson, Adam, and Hinton, Geoffrey E. The Toronto face dataset. Technical Report UTML TR 2010-001, U. Toronto, 2010.

Vincent, Pascal, Larochelle, Hugo, Bengio, Yoshua, and Manzagol, Pierre-Antoine. Extracting and composing robust features with denoising autoencoders. In *ICML 2008*, 2008.