Stochastic Back-propagation and Variational Inference in Deep Latent Gaussian Models

Danilo Jimenez Rezende, Shakir Mohamed, Daan Wierstra
{danilo, shakir, daan}@deepmind.com
DeepMind Technologies, London

Abstract

We marry ideas from deep neural networks and approximate Bayesian inference to derive a generalised class of deep, directed generative models, endowed with a new algorithm for scalable inference and learning. Our algorithm introduces a recognition model to represent approximate posterior distributions, and that acts as a stochastic encoder of the data. We develop stochastic back-propagation – rules for back-propagation through stochastic variables – and use this to develop an algorithm that allows for joint optimisation of the parameters of both the generative and recognition model. We demonstrate on several real-world data sets that the model generates realistic samples, provides accurate imputations of missing data and is a useful tool for high-dimensional data visualisation.

1. Introduction

There is an immense effort in machine learning and statistics to develop accurate and scalable probabilistic models of data. Such models are called upon whenever we are faced with tasks requiring probabilistic reasoning, such as prediction, missing data imputation and uncertainty estimation; or for those tasks requiring the ability to quickly generate samples from the model, allowing for confabulation and simulation-based analysis used in many scientific fields such as genetics, robotics and control.

Recent efforts to develop generative models have focused on directed models, since samples are easily obtained by ancestral sampling from the generative process. Directed models such as belief networks and similar latent variable models (Dayan et al., 1995; Frey, 1996; Saul et al., 1996; Bartholomew & Knott, 1999; Uria et al., 2013; Gregor et al., 2013) can be easily sampled from, but in most cases, efficient inference algorithms have remained elusive. These efforts, combined with the demand for accurate probabilistic inferences and fast simulation lead us to seek generative models that are i) *deep*, since hierarchical structure allows us to capture higher moments of the data, ii) are *non-linear*, allowing for complex structure in the data to be modelled, iii) allow for *fast sampling* of fantasy data from the inferred model, and iv) are *tractable and scalable* to large amounts of data.

We meet these desiderata by introducing a class of deep, directed generative models with Gaussian latent variables at each layer. To allow for efficient and tractable inference, we also introduce a corresponding recognition model, which can be interpreted as a stochastic encoder of the data, and represents an approximate posterior distribution. For the generative model, we derive the objective function for optimisation using variational principles; for the recognition model, we specify its structure and regularisation by exploiting recent advances in deep learning. Using this construction, we can train the entire model by a modified gradient back-propagation, which allows for optimisation of the parameters of both the generative and recognition models jointly.

We build upon the large body of prior work (contextualised in section 5) and make the following contributions:

- We marry ideas from deep neural networks and probabilistic latent variable modelling to derive a general class of deep, non-linear latent Gaussian models (section 2).
- We present a new approach for scalable variational inference that allows for joint optimisation of both variational and model parameters by exploiting the properties of latent Gaussian distributions and gradient back-propagation (sections 3 and 4).

- We provide a comprehensive and systematic evaluation of the model demonstrating its applicability to problems in simulation, visualisation, prediction and missing data imputation (section 6).

2. Deep Latent Gaussian Models

Deep latent Gaussian models (DLGM) are a general class of deep directed graphical models that consist of Gaussian latent variables at each layer of a processing hierarchy. The model consists of L layers of latent variables. To generate a sample from the model, we begin at the top-most layer (L) by drawing from a Gaussian distribution. The activation $\mathbf{h}_l$ at any lower layer is formed by a non-linear transformation of the layer above $\mathbf{h}_{l+1}$, perturbed by Gaussian noise. We then generate observations $\mathbf{v}$ by sampling from the likelihood using the activation $\mathbf{h}_1$. This process is described graphically in figure 1.

This generative process is described by the following equations:

$$\begin{aligned}
\boldsymbol{\xi}_l &= \mathcal{N}(\boldsymbol{\xi}|\mathbf{0},\mathbf{I}), \quad l = 1,\ldots,L && (1)\\
\mathbf{h}_L &= \mathbf{G}_L\boldsymbol{\xi}_L, && (2)\\
\mathbf{h}_l &= T_l(\mathbf{h}_{l+1}) + \mathbf{G}_l\boldsymbol{\xi}_l, \quad l = 1\ldots L-1 && (3)\\
\mathbf{v} &\sim \pi(\mathbf{v}|T_0(\mathbf{h}_1)), && (4)
\end{aligned}$$

where $\boldsymbol{\xi}_l$ are Gaussian variables. The transformations T_l are represented by multi-layer perceptrons (MLPs) and $\mathbf{G}_l$ are matrices. At the visible layer, the data is generated from any appropriate distribution $\pi(\mathbf{v}|\cdot)$ whose parameters are specified by a transformation of the first latent layer. This allows us to deal with data that may be of any type, such as binary, categorical, counts, real-values, or a heterogeneous combination of these data types. We typically make use of distributions in the exponential family, such as the Bernoulli distribution for binary data, but we are not restricted to this set of likelihood functions.

The joint probability distribution of this model can be expressed in two equivalent ways:

$$p(\mathbf{v},\mathbf{h}) = p(\mathbf{v}|\mathbf{h}_1)p(\mathbf{h}_L)p(\boldsymbol{\theta}^g)\prod_{l=1}^{L-1} p_l(\mathbf{h}_l|\mathbf{h}_{l+1}), \tag{5}$$

$$p(\mathbf{v},\boldsymbol{\xi}) = p(\mathbf{v}|\mathbf{h}_1(\boldsymbol{\xi}_{1\ldots L}))p(\boldsymbol{\theta}^g)\prod_{l=1}^{L}\mathcal{N}(\boldsymbol{\xi}_l). \tag{6}$$

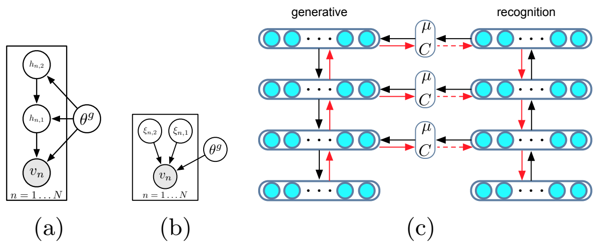


Figure 1. (a) Graphical model for deep latent Gaussian models. (b) Graphical model corresponding to the equivalent representation (6). (c) The computational diagram indicating the connectivity of the generative and recognition models. Black arrows indicate the forward pass of sampling from the recognition and generative models, and red arrows indicate the backward pass for gradient computation. The dashed arrows indicates points where stochastic rather than deterministic back-propagation is used.

A graphical model corresponding to these two views is shown in figure 1(a),(b). The conditional distribution $p(\mathbf{h}_l|\mathbf{h}_{l+1})$ is a distribution implicitly defined by equation (3) and are Gaussian distributions with mean $\mu_l = T_l(\mathbf{h}_{l+1})$ and covariance $\mathbf{S}_l = \mathbf{G}_l\mathbf{G}_l^\top$. Equation (6) makes explicit that this generative model works by applying a highly non-linear transformation to a spherical Gaussian distribution such that the transformed distribution tries to match the empirical distribution. We discuss this in more detail in appendix A. Throughout the paper we refer to the set of parameters in T_l and $\mathbf{G}_l$ by $\boldsymbol{\theta}^g$. We adopt a weak Gaussian prior over $\boldsymbol{\theta}^g$, $p(\boldsymbol{\theta}^g) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0},\kappa\mathbf{I})$.

This specification for deep latent Gaussian models generalises a number of well known models. When we have only one layer of latent variables and use a linear mapping $T(\cdot)$, we recover *factor analysis* (Bartholomew & Knott, 1999) – more general mappings allow for a non-linear factor analysis. When the mappings are of the form $T_l(\mathbf{h}) = \mathbf{A}_l f(\mathbf{h}) + \mathbf{b}_l$, for simple element-wise non-linearities f such as the probit function or the rectified linearity, we recover the *non-linear Gaussian belief network* (Frey & Hinton, 1999). We describe the relationship to other existing models in section 5. Given this specification, our key task is to develop a method for tractable inference. A number of approaches are known and currently used, in particular algorithms based on mean-field variational EM (Beal, 2003), the wake-sleep algorithm (Dayan, 2000), policy-gradient methods such as REINFORCE (Williams, 1992), or stochastic gradient methods (Hoffman et al., 2012). We develop an alternative to these inference algorithms that overcomes many of their limitations, and that is both scalable and efficient.

3. Stochastic Back-propagation

The key tool we develop to allow for efficient inference is the set of rules to allow for gradient computations in models with random variables. We develop these identities and refer to them as the rules for stochastic back-propagation.

Gradient descent methods in latent variable models typically require computations of the form: $\nabla_\theta \mathbb{E}_{q_\theta}[f(\boldsymbol{\xi})]$, where θ is a set of parameters of a distribution $q_\theta(\boldsymbol{\xi})$ and f is a loss function, which we assume to be integrable and smooth. Of special interest here are cases where the distribution q is a K-dimensional Gaussian $\mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\mu}, \mathbf{C})$ with parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}, \mathbf{C}\}$. In this setting, gradients can be computed using the Gaussian gradient identities:

$$\nabla_{\mu_i} \mathbb{E}_{\mathcal{N}(\mu,C)}[f(\boldsymbol{\xi})] = \mathbb{E}_{\mathcal{N}(\mu,C)}[\nabla_{\xi_i} f(\boldsymbol{\xi})] \quad (7)$$

$$\nabla_{C_{ij}} \mathbb{E}_{\mathcal{N}(\mu,C)}[f(\boldsymbol{\xi})] = \frac{1}{2}\mathbb{E}_{\mathcal{N}(\mu,C)}\left[\nabla^2_{\xi_i,\xi_j} f(\boldsymbol{\xi})\right], \quad (8)$$

which are due to the theorems by Bonnet (1964) and Price (1958), respectively. This allows us to write a general rule for Gaussian gradient computation.

Gaussian back-propagation (GBP). Combining equations (7) and (8) and using the chain rule we can derive the full gradient

$$\nabla_\theta \mathbb{E}_{\mathcal{N}(\mu,C)}[f(\boldsymbol{\xi})] = \mathbb{E}_{\mathcal{N}(0,I)}\left[\mathbf{g}^\top \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\theta}} + \tfrac{1}{2}\mathrm{Tr}\left(\mathbf{H}\frac{\partial \mathbf{C}}{\partial \boldsymbol{\theta}}\right)\right], \quad (9)$$

where $\mathbf{g}$ and $\mathbf{H}$ are the gradient and the Hessian of the function $f(\boldsymbol{\xi})$, respectively. Equation (9) can be interpreted as a modified back-propagation rule for Gaussian distributions that takes into account the gradients through the mean and covariance and that reduces to the standard back-propagation rule when $\mathbf{C}$ is constant. Unfortunately this rule requires knowledge of the Hessian matrix of $f(\boldsymbol{\xi})$, which has an algorithmic complexity $O(K^3)$. For inference in DLGMs, we later introduce an unbiased though higher variance estimator that requires only quadratic complexity.

We can also derive general back-propagation rules for q-distributions other than the Gaussian. Such rules can be derived in two ways:

1. Using the product rule for integrals. For many exponential family distributions, it is possible to use the product rule for integrals to express the gradient with respect to the parameters (mean or natural) as an expectation of gradients with respect to the random variables itself:

$$\nabla_\theta \mathbb{E}_{p(\xi|\theta)}[f(\xi)] = -\mathbb{E}_{p(\xi|\theta)}[\nabla_\xi[B(\xi)f(\xi)]] \quad (10)$$

We can thus reduce the problem to searching for a suitable transformation function $B(\xi)$ that allows us to compute the required derivatives directly. This approach can be used to derive rules for many other distributions e.g., the Gaussian, inverse Gamma, log-Normal and Wald distributions. We discuss this in more detail in appendix B.

2. Using suitable co-ordinate transformations. We can also derive stochastic back-propagation rules for any distribution that can be written as an smooth, invertible transformation of a canonical base distribution. For example, any Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$ can be obtained as transformation of a spherical Gaussian $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, using the transformation $y = \boldsymbol{\mu} + \mathbf{R}\boldsymbol{\epsilon}$ and $\mathbf{C} = \mathbf{R}\mathbf{R}^\top$. This co-ordinate transformation allows the gradient of the expectation with respect to $\mathbf{R}$ to be written as:

$$\begin{aligned}\nabla_R \mathbb{E}_{\mathcal{N}(\mu,C)}[f(\boldsymbol{\xi})] &= \nabla_R \mathbb{E}_{\mathcal{N}(0,I)}[f(\boldsymbol{\mu} + \mathbf{R}\boldsymbol{\epsilon})] \\ &= \mathbb{E}_{\mathcal{N}(0,I)}\left[\boldsymbol{\epsilon}\mathbf{g}^\top\right], \end{aligned} \quad (11)$$

where $\mathbf{g}$ is the gradient of f evaluated at $\boldsymbol{\mu} + \mathbf{R}\boldsymbol{\epsilon}$ and provides a lower-cost alternative to Price's theorem (8). The estimator (11) will in general have higher variance than the estimators based on (8). A short analysis of the variance of these estimators is discussed section 5 and in appendix C.

This approach can be more general than the first approach, and such transformations are well known for many distributions, especially those with a self-similarity property or location-scale formulations, such as the Gaussian, Student's t-distribution, the class of stable distributions, and the class of generalised extreme value distributions. We provide further discussion in appendix B.

4. Scalable Inference in DLGMs

4.1. Free Energy Objective

To perform inference in DLGMs we integrate out the effect of the latent variables, requiring us to compute the integrated or marginal likelihood. In general, this will be an intractable integration and instead we optimise a lower bound on the marginal likelihood. We introduce an approximate posterior distribution q and applying Jensen's inequality, following the variational principle (Zemel, 1993; Beal, 2003) to obtain:

$$\begin{aligned}\log p(\mathbf{V}) &= \log \int p(\mathbf{V}|\boldsymbol{\xi}, \boldsymbol{\theta}^g) p(\boldsymbol{\xi}, \boldsymbol{\theta}^g) d\boldsymbol{\xi} \\ &= \log \int \frac{q(\boldsymbol{\xi})}{q(\boldsymbol{\xi})} p(\mathbf{V}|\boldsymbol{\xi}, \boldsymbol{\theta}^g) p(\boldsymbol{\xi}, \boldsymbol{\theta}^g) d\boldsymbol{\xi} \quad (12) \\ &\geq \mathcal{L}(\mathbf{V}) = -D_{KL}[q(\boldsymbol{\xi}) \| p(\boldsymbol{\xi})] + \mathbb{E}_q\left[\log p(\mathbf{V}|\boldsymbol{\xi}, \boldsymbol{\theta}^g) p(\boldsymbol{\theta}^g)\right].\end{aligned}$$

We use the variational free energy, defined as $\mathcal{F}(\mathbf{V}) = -\mathcal{L}(\mathbf{V})$, as our objective function. This objective consists of two terms: the first is the KL-divergence between the variational distribution and the prior distribution (which acts a regulariser), and the second is a reconstruction error. We refer to the approximate posterior distribution $q(\boldsymbol{\xi})$ as a *recognition model*, and its design is important for the success of such variational methods.

We use a variational distribution $q(\boldsymbol{\xi})$ that factorises across the L layers, but not necessarily within a layer:

$$q(\boldsymbol{\xi}|\mathbf{V}, \boldsymbol{\theta}^r) = \prod_{n=1}^{N} \prod_{l=1}^{L} \mathcal{N}\left(\boldsymbol{\xi}_{n,l} | \boldsymbol{\mu}_l(\mathbf{v}_n), \mathbf{C}_l(\mathbf{v}_n)\right), \quad (13)$$

where $\boldsymbol{\mu}_l(\mathbf{v})$ and $\mathbf{C}_l(\mathbf{v})$ are generic maps represented by MLPs corresponding to the mean vector and covariance matrix of the units in layer l respectively. We refer to the parameters of the recognition model $q(\boldsymbol{\xi}|\mathbf{v})$ as $\boldsymbol{\theta}^r$.

For a Gaussian prior and a Gaussian recognition model, the KL term in the free energy (12) can be computed analytically and the free energy becomes:

$$\begin{aligned}\mathcal{F}(\mathbf{V}) = &-\sum_n \mathbb{E}_q\left[\log p(\mathbf{v}_n|\mathbf{h}(\boldsymbol{\xi}_n))\right] + \tfrac{1}{2\kappa}\|\boldsymbol{\theta}^g\|^2 \\ &+ \frac{1}{2}\sum_{n,l}\left[\|\boldsymbol{\mu}_{n,l}\|^2 + \mathrm{Tr}(\mathbf{C}_{n,l}) - \log|\mathbf{C}_{n,l}| - 1\right],\end{aligned} \quad (14)$$

where $\mathrm{Tr}(\mathbf{C})$ and $|\mathbf{C}|$ indicate the trace and the determinant of the covariance matrix $\mathbf{C}$, respectively.

4.2. Parameterising the Recognition Covariance

There are a number of approaches for parameterising the covariance matrix of the recognition model. Maintaining a full covariance matrix $\mathbf{C}$ in equation (14) would entail an algorithmic complexity $O(LK^3)$ where K is the number of latent variables per layer and L is the number of latent layers.

The simplest approach is to use a diagonal covariance matrix $\mathbf{C} = \mathrm{diag}(\mathbf{d})$, where $\mathbf{d}$ is a K-dimensional vector. This approach is appealing since it allows for linear-time computation and sampling, but only allows for axis-aligned posterior distributions.

We can improve upon the diagonal approximation by considering a structured covariance parameterised by two independent vectors $\mathbf{d}$ and $\mathbf{u}$. We parameterise the precision matrix $\mathbf{C}^{-1}$ (Magdon-Ismail & Purnell, 2010):

$$\mathbf{C}^{-1} = \mathbf{D} + \mathbf{u}\mathbf{u}^T, \quad (15)$$

where $\mathbf{D} = \mathrm{diag}(\mathbf{d})$. This representation allows for arbitrary rotations of the Gaussian distribution along one principal direction, with relatively few additional parameters.

By application of the matrix inversion lemma, we obtain the covariance matrix in terms of $\mathbf{d}$ and $\mathbf{u}$ as:

$$\begin{aligned}&\mathbf{C} = \mathbf{D}^{-1} + \eta\mathbf{D}^{-1}\mathbf{u}\mathbf{u}^T\mathbf{D}^{-1}, \qquad \eta = \tfrac{1}{\mathbf{u}^T\mathbf{D}^{-1}\mathbf{u}+1}, \\ &\log|\mathbf{C}| = \log\eta - \log|\mathbf{D}|.\end{aligned} \quad (16)$$

This allows both the trace $\mathrm{Tr}(\mathbf{C})$ and the $\log|\mathbf{C}|$ needed in the computation of the Gaussian KL to be computed in $O(KL)$ time.

In addition, we can factorise the covariance matrix as the product of two matrices $\mathbf{C} = \mathbf{R}\mathbf{R}^T$. $\mathbf{R}$ is a square matrix of the same size as $\mathbf{C}$ and can be computed directly in terms of the vectors $\mathbf{d}$ and $\mathbf{u}$. One solution for $\mathbf{R}$ is:

$$\mathbf{R} = \mathbf{D}^{-\frac{1}{2}} - \left[\frac{1-\sqrt{\eta}}{\mathbf{u}^T\mathbf{D}^{-1}\mathbf{u}}\right]\mathbf{D}^{-1}\mathbf{u}\mathbf{u}^T\mathbf{D}^{-\frac{1}{2}}. \quad (17)$$

Due to this structure, the product of $\mathbf{R}$ with an arbitrary vector can be computed in $O(K)$ without having to compute $\mathbf{R}$ explicitly. This allows us to also sample efficiently from this low-rank Gaussian, since any Gaussian random variable $\boldsymbol{\xi}$ with mean $\boldsymbol{\mu}$ and covariance matrix $\mathbf{C} = \mathbf{R}\mathbf{R}^T$ can be written as $\boldsymbol{\xi} = \boldsymbol{\mu} + \mathbf{R}\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a standard Gaussian variate.

4.3. Gradients of the Free Energy

We wish to minimise the free energy $\mathcal{F}(\mathbf{V})$ (14) using stochastic gradient descent methods. For this, we need efficient estimators of the gradients of all terms in (14) with respect to both the parameters of the generative model $\boldsymbol{\theta}^g$ and the recognition model $\boldsymbol{\theta}^r$.

The gradients with respect to the jth generative parameter θ_j^g can computed using:

$$\nabla_{\theta_j^g}\mathcal{F}(\mathbf{V}) = -\mathbb{E}_q\left[\nabla_{\theta_j^g}\log p(\mathbf{V}|\mathbf{h})\right] + \frac{1}{\kappa}\theta_j^g. \quad (18)$$

An unbiased estimator of $\nabla_{\theta_j^g}\mathcal{F}(\mathbf{V})$ is obtained by approximating (18) with a small number of samples from q (or even a single sample).

To obtain gradients with respect to the recognition parameters $\boldsymbol{\theta}^r$, we use the rules for Gaussian backpropagation developed in section 3. To address the complexity of the Hessian in the general rule (9), we use the co-ordinate transformation for the Gaussian to write the gradient with respect to the matrix $\mathbf{R}$ instead of the covariance $\mathbf{C}$ (recalling $\mathbf{C} = \mathbf{R}\mathbf{R}^\top$) derived in equation (11), where derivatives are computed for the function $f(\boldsymbol{\xi}) = \log p(\mathbf{v}|\mathbf{h}(\boldsymbol{\xi}))$.

The gradients of $\mathcal{F}(\mathbf{v})$ with respect to the variational mean $\boldsymbol{\mu}_l(\mathbf{v})$ and the matrices $\mathbf{R}_l(\mathbf{v})$ are given by

$$\nabla_{\mu_l}\mathcal{F}(\mathbf{v}) = -\mathbb{E}_q\left[\nabla_{\boldsymbol{\xi}_l}\log p(\mathbf{v}|\mathbf{h}(\xi))\right] + \boldsymbol{\mu}_l \tag{19}$$

$$\begin{aligned}\nabla_{R_{l,i,j}}\mathcal{F}(\mathbf{v}) &= -\frac{1}{2}\mathbb{E}_q\left[\epsilon_{l,j}\nabla_{\xi_{l,i}}\log p(\mathbf{v}|\mathbf{h}(\boldsymbol{\xi}))\right] \\ &+ \frac{1}{2}\nabla_{R_{l,i,j}}\left[\operatorname{Tr}\mathbf{C}_{n,l} - \log|\mathbf{C}_{n,l}|\right], \end{aligned}\tag{20}$$

where the gradients $\nabla_{R_{l,i,j}}\left[\operatorname{Tr}\mathbf{C}_{n,l} - \log|\mathbf{C}_{n,l}|\right]$ are computed by back-propagation.

Unbiased estimators of the gradients (19) and (20) are then obtained jointly by sampling from the recognition model $\xi \sim q(\xi|\mathbf{v})$ (bottom-up pass) and updating the values of the generative model layers using equation (3) (top-down pass).

Finally the gradients $\nabla_{\boldsymbol{\theta}_j^r}\mathcal{F}(\mathbf{v})$, obtained from equations (19) and (20), are:

$$\nabla_{\boldsymbol{\theta}^r}\mathcal{F}(\mathbf{v}) = \nabla_{\boldsymbol{\mu}}\mathcal{F}(\mathbf{v})^\top\frac{\partial\boldsymbol{\mu}}{\partial\boldsymbol{\theta}^r} + \operatorname{Tr}\left(\nabla_{\mathbf{R}}\mathcal{F}(\mathbf{v})\frac{\partial\mathbf{R}}{\partial\boldsymbol{\theta}^r}\right). \tag{21}$$

We can now use the gradients (18) and (21) to descend the free-energy surface with respect to both the generative and recognition parameters in a single optimisation step.

4.4. Algorithm Summary and Complexity

Figure 1 (c) shows the flow of computation in the model. Our algorithm proceeds by performing a forward pass (black arrows): consisting of bottom-up (recognition) phase and a top-down (generation phase), which updates the hidden activations of the recognition model and parameters of any Gaussian distributions; and a backward pass (red arrows) in which gradients are computed using the appropriate back-propagation rule for deterministic and stochastic layers.

We take a gradient descent step using:

$$\Delta\theta^{g,r} = -\Gamma^{g,r}\nabla_{\theta^{g,r}}\mathcal{F}(\mathbf{V}), \tag{22}$$

where $\Gamma^{g,r}$ are diagonal pre-conditioning matrices computed using the RMSprop heuristic[1]. The learning procedure is summarised in algorithm (1).

The computational complexity for producing S samples from the generative model is $O(SL\bar{K}^2)$ where $\bar{K}$ is the average number of neurons per layer, L is the number of layers (counting together deterministic and stochastic layers). The computational complexity per training sample during training is $O(L\bar{K}^2)$, and is the same as that of an auto-encoder with a decoder and encoder of same size as the generative and recognition models, respectively.

Algorithm 1 Learning in DLGMs

```
while hasNotConverged() do
    $\mathbf{V} \leftarrow$ getMiniBatch()
    $\boldsymbol{\xi}_n \sim q(\xi_n|\mathbf{v}_n)$ (bottom-up pass) eq. (13)
    $\mathbf{h} \leftarrow \mathbf{h}(v\xi)$ (top-down pass) eq. (3)
    updateGradients() eqs (19), (18), (20)
    $\boldsymbol{\theta}^{g,r} \leftarrow \Delta\boldsymbol{\theta}^{g,r}$
end while
```

5. Related Work

Directed Graphical Models. There is a broad existing literature on directed graphical models with depth. Our model is directly related to non-linear Gaussian belief networks (NLGNBN) (Frey & Hinton, 1999), which also use latent Gaussian distributions throughout a multi-layer hierarchy, but whose means are simple non-linear transformations of higher layers. Sigmoid belief networks (SBN) (Saul et al., 1996) are closely related and are models with Bernoulli variables at every layer. Both these models are trained by a mean-field variational EM algorithm. More recently, Gregor et al. (2013) described Deep Auto-regressive Networks (DARN), which also form a directed graphical model that uses auto-regressive Bernoulli distributions at each layer. The Gaussian process latent variable model (GPLVM) (Lawrence, 2005) is the non-parametric analogue of our model, and employs Gaussian process priors over the non-linear functions between each layer. Some of the best results using directed models are provided by the neural autoregressive density estimator (NADE) (Larochelle & Murray, 2011; Uria et al., 2013), which uses function approximation to model conditional distributions within a directed acyclic graph. We will also see this performance on the benchmark tests we present, but major drawbacks exist, such as its inability to allow for deep representations and difficulties in extending to locally connected models (e.g., through the use of convolutional layers) that allow for high-dimensional scaling.

Relation to denoising auto-encoders. Denoising auto-encoders (DAE) (Vincent et al., 2010) introduce a random corruption to the encoder network and at-

[1] Described by G. Hinton, *RMSprop: Divide the gradient by a running average of its recent magnitude*, in Coursera online course: Neural networks for machine learning, lecture 6e, 2012.

tempt to minimize the expected reconstruction error under this corruption noise with additional regularisation terms. Bengio et al. (2013) show how these models can be used as generative models, but since the generative process underlying the denoising auto-encoder is unknown, simulation from the model requires a slow Markov chain sampling procedure. In the model we describe, the recognition distribution $q(\boldsymbol{\xi}|\mathbf{v})$ can be interpreted as a stochastic encoder in the DAE setting. We can readily see the correspondence between the expression for the free energy (12) and the reconstruction error and regularization terms used in denoising auto-encoders (c.f. equation (4) of Bengio et al. (2013)).

It is possible to now see denoising auto-encoders in the framework for variational learning in latent variable models. The key difference is that the form of encoding 'corruption' and regularisation terms used in our model have been derived from the variational principle to provide a strict bound on the data log-likelihood of a known directed generative model and that allows for easy generation of samples.

Alternative Gradient Estimates. Other approaches for gradient estimation are typically used in the literature. The most general approaches are policy-gradient methods such as REINFORCE (Williams, 1992) that are simple to implement and applicable to both discrete and continuous models. This estimator can be written as:

$$\nabla_\theta \mathbb{E}_p[f(\xi)] = \mathbb{E}_p[(f(\xi) - b)\nabla_\theta \log p(\xi|\theta)], \qquad (23)$$

where b is a baseline typically chosen to reduce the variance of the estimator; we use draws $\xi \sim p(\xi|\theta)$.

Unfortunately the variance of (23) scales poorly with the number of random variables (Dayan et al., 1995). To see this limitation, consider functions of the form $f(\xi) = \sum_{i=1}^{D} f(\xi_i)$, where each individual term has a bounded variance, i.e., $\mathrm{Var}[f(\xi_i)] \leq \kappa \forall i$ for some $\kappa > 0$, and consider independent or weakly correlated random variables. Given these assumptions we have the following bounds for the variances of (7) and REINFORCE (23):

$$\begin{aligned} &\text{GBP:} && \mathrm{Var}[\nabla_\xi f(\xi)] \leq \kappa \\ &\text{REINFORCE:} && \mathrm{Var}\left[\tfrac{(\xi-\mu)}{\sigma^2}(f(\xi) - \mathbb{E}[f(\xi)])\right] \leq D\kappa. \end{aligned}$$

Thus, the REINFORCE estimator has the undesirable property that its variance scales linearly with the number of independent random variables in the target function, while the variance of GBP is bounded by a constant.

Any correlation between the different terms in $f(\xi)$ would reduce the absolute value of the variance (5). The assumption of weakly correlated terms is relevant for variational learning in larger generative models where variables tend to depend strongly only on other variables in their Markov blanket (i.e. only neighbouring nodes in the graphical model tend to display strong correlations). Moreover, due to independence assumptions and structure in the variational distribution, the resulting free energies are often summations over weakly correlated or independent terms, further supporting this view. We provide a second view on the issue in appendix C.

Another very general alternative to REINFORCE is the wake-sleep algorithm (Dayan et al., 1995). The wake-sleep algorithm fails to optimise a single consistent objective function and there is thus no guarantee that it leads to a decrease of the free energy (12). But it has been shown to work well in some small practical examples. In figure 2(c) we provide a comparison of the performance achieved by our model and the wake-sleep algorithm.

Stochastic back-propagation in other contexts. The key theorems for the Gaussian stochastic back-propagation were first exploited by Opper & Archambeau (2009) for variational learning in Gaussian process regression, and subsequently by Graves (2011) for learning the parameters of large neural networks.

Opper & Archambeau (2009) make use of an unconditional variational approximation, which typically requires several iterations for every visible pattern $\mathbf{v}$ in order to converge to a local minimum of the free energy. In contrast, we use a parametric recognition model (eq. (13)) which can produce a one-shot sample from the approximate posterior distribution (similar to the procedure in Dayan et al. (1995); Dayan (2000)). Using a conditional, parametric recognition model also provides a cheap probabilistic encoding of the dataset and provides a framework for treating generative models and classes of auto-encoders on the same principled ground (as described above). Recently, Kingma & Welling (2013) also make the connection between stochastic back-propagation, generative auto-encoders and variational inference that we describe here. This work was developed simultaneously with ours and provides an additional perspective on the use and derivation of stochastic back-propagation rules.

6. Results

Generative models, such as the deep latent Gaussian model that we focus on, have a number of applications in simulation, prediction, data visualisation, missing data imputation and other forms of probabilistic reasoning. We describe the testing methodology we use and present results on a number of these tasks.

6.1. Testing Methodology

We use training data of various types including binary and count-based data sets. In all cases, we train using mini-batches, which requires the introduction of scaling terms in the free energy objective function (14) in order to maintain the correct scale between the prior over the parameters and the remaining terms (Ahn et al., 2012; Welling & Teh, 2011). We make use of the objective:

$$\overline{\mathcal{F}(\mathbf{V})} = -\lambda \sum_n \mathbb{E}_q \left[\log p(\mathbf{v}_n|\mathbf{h}(\boldsymbol{\xi}_n))\right] + \tfrac{1}{2\kappa}\|\boldsymbol{\theta}^g\|^2$$
$$+ \frac{\lambda}{2} \sum_{n,l} \left[\|\boldsymbol{\mu}_{n,l}\|^2 + \mathrm{Tr}(\mathbf{C}_{n,l}) - \log|\mathbf{C}_{n,l}| - 1\right], \quad (24)$$

where n is an index over observations in the mini-batch and λ is equal to the ratio between the data-set and the mini-batch size. At each iteration, a random mini-batch of size 200 observations is chosen.

All parameters of the model were initialized using samples from a Gaussian distribution with mean zero and variance 1×10^6; the prior variance of the parameters was $\kappa = 1 \times 10^6$. We compute the marginal likelihood on the test data by importance sampling using samples from the recognition model; we describe our estimator in appendix D.

Since the recognition model is parameterised by a deep neural network, careful regularisation is needed to ensure that it provides useful inferences for unseen data. We regularise by introducing additional noise to the recognition model, specifically, bit-flip or drop-out noise at the input layer and small additional Gaussian noise to samples generated by the recognition model. We also use rectified non-linearities for any hidden layers.

6.2. Analysing the Approximate Posterior

The specification of the recognition model affects how well we are able to capture the statistics of the data and the accuracy and efficiency of the learning.

We examine the quality of the approximate posterior distribution learnt by the recognition model in comparison to the true posterior distribution by training a deep latent Gaussian model on the MNIST data set of handwritten images. The images are of size 28×28, and we use the binarised data set from Larochelle & Murray (2011).

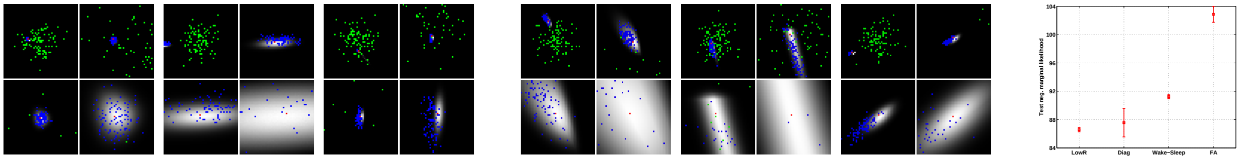
We use sampling to evaluate the true posterior distribution for a number of MNIST digits under the diagonal and the structured covariance parameterisation of the recognition model, described in section 4.2. We visualise the posterior distribution for a model with two Gaussian latent variables in figure 2. The true posterior distribution is shown by the grey regions and was computed by importance sampling with a large number of particles aligned in a grid between -5 and 5. In figure 2(a) we see that many posterior distributions are elliptical or spherical in shape and thus, it is reasonable to assume that they can be well described by a Gaussian approximation. Samples from the prior (shown in green) are spread widely over the space and very few samples fall in the region of significant posterior mass, explaining the inefficiency of estimation methods that rely on samples from the prior. Samples from the recognition model (shown in blue) are concentrated on the posterior mass, indicating that the recognition model has learnt the correct posterior statistics and should lead to efficient learning.

In figure 2(a) we see that samples from the recognition model are aligned to the axis and cannot capture any correlation in the posterior. Using the low-rank covariance model, we see in figure 2(b) that we are able to capture the posterior correlation. Not all posteriors are Gaussian in shape, but the recognition places mass in the best location possible to provide a reasonable approximation. This aspect emphasises the importance of posterior approximation, and one we continue to investigate. The performance in terms of test log-likelihood on the MNIST data is shown for four algorithms on the same model architecture. We compare factor analysis (FA), the wake-sleep algorithm, and our approach using a diagonal and low-rank covariance.

6.3. Simulation and prediction

We evaluate the performance of a three layer latent Gaussian model on the MNIST data set. The model consists of two deterministic layers with 200 hidden units and a stochastic layer of 200 latent variables. We use mini-batches of 200 observations and trained the model using stochastic back-propagation. Samples from this model are shown in figure 3. We also compare the test log-likelihood to a large number of existing approaches in table 1. We used exactly the data set used in Uria et al. (2013) and quote the log-likelihoods in the lower part of the table from this work. These results show that we are able to compete with some of the best models currently available. The generated digits also match the true data well and visually appear as good as some of the best visualisations from these competing approaches.

We also analysed the performance of our model on



(a) Diagonal covariance (b) Low-rank covariance (c) Performance

Figure 2. (a),(b) Analysis of the exact posterior vs. our recognition model for 6 MNIST digits. The grey regions show the true posterior distribution, green markers indicate samples from the prior distribution, blue markers are samples from the recognition model, and the red marker indicates the MAP estimate. Within each image we show four views of the same posterior, zooming in on the region of high posterior mass. (c) Comparison of test log likelihoods for different inference algorithms.

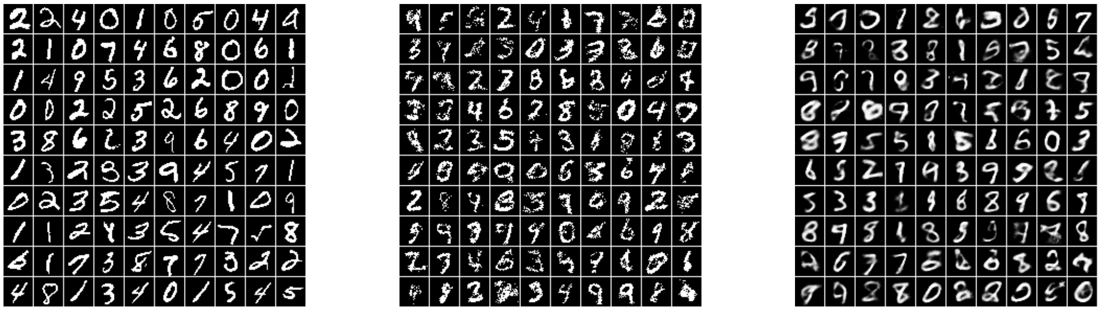


Figure 3. Performance on the MNIST dataset. Top left: Samples from the training data. Top right: Samples from the learned model. Bottom right: sampled pixel probabilities

Table 1. Comparison of negative log-probabilities on the test set for the binarised MNIST data.

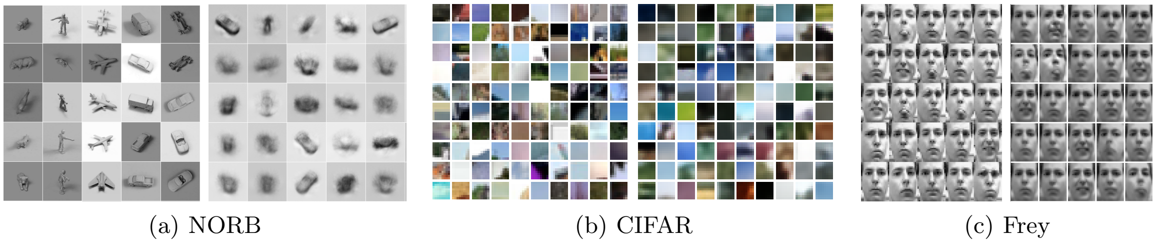
Model	$-\ln p(\mathbf{v})$
Factor Analysis	106.00
NLGBN (Frey & Hinton, 1999)	95.80
Wake-Sleep (Dayan, 2000)	91.3
DLGM diagonal covariance	87.30
DLGM rank-one covariance	**86.60**
Results below from Uria et al. (2013)	
MoBernoullis K=10	168.95
MoBernoullis K=500	137.64
RBM (500 h, 25 CD steps) approx.	86.34
DBN 2hl approx.	**84.55**
NADE 1hl (fixed order)	88.86
NADE 1hl (fixed order, RLU, minibatch)	88.33
EoNADE 1hl (2 orderings)	90.69
EoNADE 1hl (128 orderings)	87.71
EoNADE 2hl (2 orderings)	87.96
EoNADE 2hl (128 orderings)	**85.10**

three high-dimensional real image data sets. The NORB object recognition data set consists of 24, 300 images that are 96×96 pixels. We use a model consisting of 1 deterministic layer of 400 hidden units and one stochastic layer of 100 latent variables. Samples produced from this model are shown in figure 4(a). The CIFAR10 natural images data set consists of 50, 000 RGB images that are 32×32 pixels, which we split into 8×8 patches. We use the same model as used for the MNIST experiment and show samples from the model in figure 4(b). The Frey faces data set consists of almost 2, 000 images of different facial expressions[2] of size 28×20 pixels. In all cases we see that the model has learnt the different image statistics and produces recognisable samples. Results of this kind are extremely promising and suggest that performance can be improved greatly by scaling the model to more realistic scenarios by considering locally connected and convolutional architectures.

6.4. Missing Data Imputation and Denoising

The generative models we describe are often used for problems in missing data imputation that form the core of applications in recommender system, bioinfor-

[2] Images of Brendan Frey. Data from `http://www.cs.nyu.edu/~roweis/data.html`


(a) NORB

(b) CIFAR

(c) Frey

Figure 4. a) Performance on the NORB dataset. Left: Samples from the training data. Right: sampled pixel means from the model. b) Performance on CIFAR10 patches. Left: Samples from the training data. Right: Sampled pixel means from the model. c) Frey faces data. Left: data samples. Right: model samples.

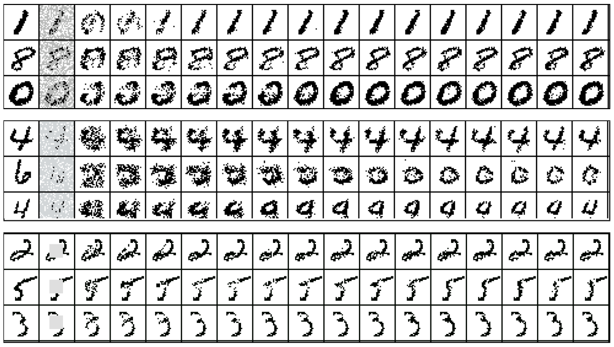


Figure 5. Imputation results on MNIST digits. The first column shows the true data. Column 2 shows pixel locations set as missing in grey. The remaining columns show imputations and denoising of the images for 15 iterations, starting left to right. Top: 60% missingness. Middle: 80% missingness. Bottom: 5x5 patch missing.

matics and experimental design. We show the ability of the model to impute missing data using the MNIST data set in figure 5. We test the imputation ability under two different missingness types (Little & Rubin, 1987): Missing-at-random (MAR), where we consider 60% and 80% of the pixels to be missing randomly, and Not Missing-at-random (NMAR), where we consider a square region of the image to be missing. The model produces very good completions in both test cases. There is uncertainty in the identity of the image. This is expected and reflected in the errors in these completions as the resampling procedure is run, and further demonstrates the ability of the model to capture the diversity of the underlying data. We do not integrate over the missing values in our imputation procedure, but use a procedure that simulates a Markov chain that we show converges to the true marginal distribution. The procedure to sample from the missing pixels given the observed pixels is explained in appendix E.

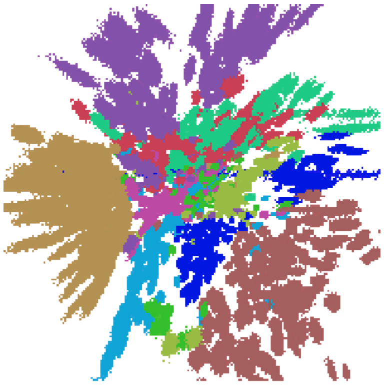


Figure 6. Two dimensional embedding of the MNIST data set. Each colour corresponds to one of the digit classes.

6.5. Data Visualisation

Latent variable models such as DLGMs are often used for visualisation of high-dimensional data sets. We project the MNIST data set to a 2-dimensional latent space and use this 2-D embedding as a visualisation of the data. A 2-dimensional embedding of the MNIST data set is shown in figure 6. The classes separate into different regions indicating that such a tool can be useful in gaining insight into the structure of high-dimensional data sets.

7. Discussion

Our algorithm generalises to a large class of models with continuous latent variables, which include Gaussian, non-negative or sparsity-promoting latent variables. For models with discrete latent variables (e.g., sigmoid belief networks), policy-gradient approaches that improve upon the REINFORCE approach remain the most general, but intelligent design is needed to control the gradient-variance in high dimensional settings.

These models are typically used with a large number of latent variables. In this setting, and under the appropriate conditions, the required expectations for inference could be well approximated by Gaussian integrals. Thus, we believe that our approach is applicable even in the high-dimensional discrete setting: we can apply the gradient estimators derived in section 4 as an approximation in high-dimensional discrete latent variable models, and potentially obtain new learning rules for these models.

An additional feature of our approach is that it can easily be combined with convolutional architectures and computation using GPUs to allow for scaling to the large-data settings we are now routinely faced with. More investigation is required to fully explore the impact of the structure of the recognition model and to allow more accurate covariance estimation. This is particularly important in the high-dimensional setting where we lack intuition regarding the characteristics of the posterior distribution. This and other extensions form the basis of much future work.

8. Conclusion

We have developed a class of general-purpose and flexible generative models with Gaussian latent variables at each layer. Our approach introduces a recognition model, which can be seen as a stochastic encoding of the data, to allow for efficient and tractable inference. We derived a lower bound on the marginal likelihood for the generative model and specified the structure and regularisation of the recognition model by exploiting recent advances in deep learning. By developing modified rules for back-propagation through stochastic layers, we derived an efficient inference algorithm that allows for joint optimisation of all parameters, i.e. parameters of the generative and recognition models. We have demonstrated on several real-world data sets that the model generates realistic samples, provides accurate imputations of missing data and can be a useful tool for high-dimensional data visualisation.

References

Ahn, S., Balan, A. K., and Welling, M. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *ICML*, 2012.

Bartholomew, D. J. and Knott, M. *Latent variable models and factor analysis*, volume 7 of Kendall's library of statistics. Arnold, 2nd edition, 1999.

Beal, M. J. *Variational Algorithms for approximate Bayesian inference*. PhD thesis, University of Cambridge, 2003.

Bengio, Y., Yao, L., Alain, G., and Vincent, P. Generalized denoising auto-encoders as generative models. pp. 1–9, 2013.

Bonnet, G. Transformations des signaux aléatoires a travers les systèmes non linéaires sans mémoire. *Annales des Télécommunications*, 19(9-10):203–220, 1964.

Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R. S. The Helmholtz machine. *Neural computation*, 7(5):889–904, September 1995.

Dayan, P. Helmholtz machines and wake-sleep learning. *Handbook of Brain Theory and Neural Network. MIT Press, Cambridge, MA*, 44(0), 2000.

Frey, B. J. Variational inference for continuous sigmoidal Bayesian networks. *6th International Workshop on Artificial Intelligence and Statistics*, 1996.

Frey, B. J. and Hinton, G. E. Variational learning in nonlinear Gaussian belief networks. *Neural Computation*, 11(1):193–213, January 1999.

Graves, A. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems 24*, pp. 2348–2356. 2011.

Gregor, K., Mnih, A., and Wierstra, D. Deep autoregressive networks. *ArXiv preprint. arXiv:1310.8499*, October 2013.

Hoffman, M., Blei, D. M., Wang, C., and Paisley, J. Stochastic variational inference. *arXiv preprint arXiv:1206.7051*, 2012.

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

Larochelle, H. and Murray, I. The neural autoregressive distribution estimator. *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, JMLR W&CP 15:29–37, 2011.

Lawrence, N. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *The Journal of Machine Learning Research*, 6:1783–1816, 2005.

Little, R. J. and Rubin, D. B. *Statistical analysis with missing data*, volume 539. Wiley New York, 1987.

Magdon-Ismail, M. and Purnell, J. T. Approximating the covariance matrix of GMMs with low-rank perturbations. pp. 300–307, 2010.

Minka, T. *A family of algorithms for approximate Bayesian inference*. PhD thesis, MIT, 2001.

Neal, R. M. Probabilistic inference using Markov Chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto, 1993.

Opper, M. and Archambeau, C. The variational Gaussian approximation revisited. *Neural computation*, 21(3):786–92, March 2009.

Price, R. A useful theorem for nonlinear devices having Gaussian inputs. *IEEE Transactions on Information Theory*, 4(2):69–72, 1958.

Rue, H., Martino, S., and Chopin, N. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(2):319–392, 2009.

Saul, L. K., Jaakkola, T., and Jordan, M. I. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research (JAIR)*, 4:61–76, 1996.

Uria, B., Murray, I., and Larochelle, H. A deep and tractable density estimator. *ArXiv preprint. arXiv:1310.1757v1*, October 2013.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *The Journal of Machine Learning Research*, 11:3371–3371–3408–3408, 2010.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 681–688, 2011.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229 – 256, 1992.

Zemel, R. S. A minimum description length framework for unsupervised learning. January 1993.

A. Additional Model Details

In equation (6) we showed an alternative form of the joint log likelihood that makes explicit that the generative model works by applying a highly non-linear transformation to a spherical Gaussian distribution $\mathcal{N}(\boldsymbol{\xi})$ such that the transformed distribution best matches the empirical distribution. We provide more details on this view here for clarity.

From the model description (3), (4), we can interpret the variables $\mathbf{h}_l$ as deterministic functions of the noise variables $\boldsymbol{\xi}_l$. This can be formally introduced as a coordinate transformation of the probability density in equation (5): we perform a change of coordinates $\mathbf{h}_l \rightarrow \boldsymbol{\xi}_l$. The density of the transformed variables $\boldsymbol{\xi}$ can be expressed in terms of the density (5) times the determinant of the Jacobian of the transformation $p(\boldsymbol{\xi}) = p(\mathbf{h}(\boldsymbol{\xi}))|\frac{\partial \mathbf{h}}{\partial \boldsymbol{\xi}}|$. Since the coordinate transformation is linear we have $|\frac{\partial \mathbf{h}}{\partial \boldsymbol{\xi}}| = |\mathbf{G}_l|$ and the distribution of $\boldsymbol{\xi}$ is:

$$p(\boldsymbol{\xi}) = p(\mathbf{h}_L)|\mathbf{G}_L| \prod_{l=1}^{L-1} |\mathbf{G}_l| p_l(\mathbf{h}_l|\mathbf{h}_{l+1}) = \prod_{l=1}^{L} |\mathbf{G}_l||S_l|^{-\frac{1}{2}} \mathcal{N}(\xi_l)$$
$$= \prod_{l=1}^{L} |G_l||G_l G_l^T|^{-\frac{1}{2}} \mathcal{N}(\boldsymbol{\xi}_l) = \prod_{l=1}^{L} \mathcal{N}(\boldsymbol{\xi}_l), \tag{25}$$

where $\mathcal{N}(\boldsymbol{\xi})$ is a Gaussian with mean zero and covariance equal to the identity matrix. Using this view, we rewrite the joint probability as in (6).

A simple recognition model that can be used, consists of a single deterministic layer and a stochastic Gaussian layer with the rank-one covariace structure and is constructed as:

$$q(\boldsymbol{\xi}|\mathbf{v}) = \mathcal{N}\left(\boldsymbol{\xi}|\boldsymbol{\mu}; (\mathrm{diag}(\mathbf{d}) + \mathbf{u}\mathbf{u}^\top)^{-1}\right) \tag{26}$$
$$\boldsymbol{\mu} = \mathbf{W}_\mu \mathbf{z} + \mathbf{b}_\mu \tag{27}$$
$$\mathbf{d} = \mathbf{W}_d \mathbf{z} + \mathbf{b}_d; \qquad \mathbf{u} = \mathbf{W}_u \mathbf{z} + \mathbf{b}_u \tag{28}$$
$$\mathbf{z} = f(\mathbf{W}_v \mathbf{v} + \mathbf{b}_v) \tag{29}$$

where the function f is a rectified non-linearity (but other non-linearities such as tanh can be used.)

B. Deriving Stochastic Back-propagation Rules

In section 3 we described two ways in which to derive stochastic back-propagation rules. We show specific examples and provide some more discussion in this section.

B.1. Using the Product Rule for Integrals

We can derive rules for stochastic back-propagation for many distributions by finding a appropriate non-linear function that allows us to express the gradient with respect to the parameters of the distribution as a gradient with respect to the random variable directly. The approach we described in the main text was:

$$\nabla_\theta \mathbb{E}_p[f(x)] = \int \nabla_\theta p(x|\theta) f(x) dx = \int \nabla_x p(x|\theta) B(x) f(x) dx$$
$$= [B(x) f(x) p(x|\theta)]_{supp(x)} - \int p(x|\theta) \nabla_x [B(x) f(x)]$$
$$= -\mathbb{E}_{p(x|\theta)}[\nabla_x [B(x) f(x)]] \tag{30}$$

where we have introduced the non-linear function $B(x)$ to allows the transformation of the gradients and have applied the product rule for integrals (rule for integration by parts) to rewrite the integral in two parts in the second line, and the supp(x) indicates that the term is evaluated at the boundaries of the support. To use this approach, we require that the density we are analysing be zero at the boundaries of the support to ensure that the first term in the second line is zero.

As an alternative, we can also write this differently and find an non-linear function of the form:

$$\nabla_\theta \mathbb{E}_p[f(x)] == -\mathbb{E}_{p(x|\theta)}[B(x) \nabla_x f(x)]. \tag{31}$$

Consider general exponential family distributions of the form:

$$p(x|\theta) = h(x) \exp(\eta(\theta)^\top \phi(x) - A(\theta)) \tag{32}$$

where $h(x)$ is the base measure, θ is the set of mean parameters of the distribution, η is the set of natural parameters, and $A(\theta)$ is the log-partition function. We can express the non-linear function in (30) using these quantities as:

$$B(x) = \frac{[\nabla_\theta \eta(\theta) \phi(x) - \nabla_\theta A(\theta)]}{[\nabla_x \log[h(x)] + \eta(\theta)^T \nabla_x \phi(x)]}. \tag{33}$$

This can be derived for a number of distributions such as the Gaussian, inverse Gamma, Log-Normal, Wald (inverse Gaussian) and other distributions. We show some of these below:

The $B(x)$ corresponding to the second formulation can also be derived and may be useful in certain situations, requiring the solution of a first order differential equation. This approach of searching for non-linear transformations leads us to the second approach for deriving stochastic back-propagation rules.

Family	θ	$B(x)$
Gaussian	$\begin{pmatrix} \mu \\ \sigma^2 \end{pmatrix}$	$\begin{pmatrix} -1 \\ \frac{(x-\mu-\sigma)(x-\mu+\sigma)}{2\sigma^2(x-\mu)} \end{pmatrix}$
Inv. Gamma	$\begin{pmatrix} \alpha \\ \beta \end{pmatrix}$	$\begin{pmatrix} \frac{x^2(-\ln x-\Psi(\alpha)+\ln\beta)}{-x(\alpha+1)+\beta} \\ (\frac{x^2}{-x(\alpha+1)+\beta})(-\frac{1}{x}+\frac{\alpha}{\beta}) \end{pmatrix}$
Log-Normal	$\begin{pmatrix} \mu \\ \sigma^2 \end{pmatrix}$	$\begin{pmatrix} -1 \\ \frac{(\ln x-\mu-\sigma)(\ln x-\mu+\sigma)}{2\sigma^2(\ln x-\mu)} \end{pmatrix}$

B.2. Using Alternative Co-ordinate Transformations

There are many distributions outside the exponential family that we would like to consider using. A simpler approach is to search for a co-ordinate transformation that allows us to separate the deterministic and stochastic parts of the distribution. We described the case of the Gaussian in section 3. Other distributions also have this property. As an example, consider the Levy distribution (which is a special case of the inverse Gamma considered above). Due to the self-similarity property of this distribution, if we draw X from a Levy distribution with known parameters $X \sim Levy(\mu, \lambda)$, we can obtain any other Levy distribution by rescaling and shifting this base distribution: $kX + b \sim Levy(k\mu + b, kc)$.

Many other distributions hold this property, allowing stochastic back-propagation rules to be determined for distributions such as the Student's t-distribution, Logistic distribution, the class of stable distributions and the class of generalised extreme value distributions (GEV). Examples of co-ordinate transformations $T(\cdot)$ and the resulsting distributions are shown below for variates X drawn from the standard distribution listed in the first column.

Std Distr.	$T(\cdot)$	Gen. Distr.
$GEV(\mu, \sigma, 0)$	$mX+b$	$GEV(m\mu+b, m\sigma, 0)$
$Exp(1)$	$\mu+\beta\ln(1+\exp(-X))$	$Logistic(\mu, \beta)$
$Exp(1)$	$\lambda X^{\frac{1}{k}}$	$Weibull(\lambda, k)$

C. Univariate variance analysis

In analysing the variance properties of many estimators, we showed the general scaling of likelihood ratio approaches in section 5. As an example to further emphasise the high-variance nature of these alternative approaches, we present a short analysis in the univariate case.

Consider a random variable $p(\xi) = \mathcal{N}(\xi|\mu, \sigma^2)$ and a simple quadratic function of the form

$$f(\xi) = c\frac{\xi^2}{2}. \tag{34}$$

For this function we immediately obtain the following variances

$$Var[\nabla_\xi f(\xi)] = c^2\sigma^2 \tag{35}$$

$$Var[\nabla_{\xi^2} f(\xi)] = 0 \tag{36}$$

$$Var[\frac{(\xi-\mu)}{\sigma}\nabla_\xi f(\xi)] = 2c^2\sigma^2 + \mu^2 c^2 \tag{37}$$

$$Var[\frac{(\xi-\mu)}{\sigma^2}(f(\xi) - \mathbb{E}[f(\xi)])] = 2c^2\mu^2 + \frac{5}{2}c^2\sigma^2 \tag{38}$$

Equations (35), (36) and (37) correspond to the variance of the estimators based on (7), (8), (11) respectively whereas equation (38) corresponds to the variance of the REINFORCE algorithm for the gradient with respect to μ.

From these relations we see that, for any parameter configuration, the variance of the REINFORCE estimator is strictly larger than the variance of the estimator based on (7). Additionally, the ratio between the variances of the former and later estimators is lower-bounded by 5/2. We can also see that the variance of the estimator based on (8) is zero for this specific function whereas the variance of the estimator based on (11) is not.

The high variance nature of the policy gradient approaches are undesirable and is magnified in multivarate settings, leading to the variance estimates that scale with the dimensionality of the latent variables that were discussed in section 5

D. Estimating the Marginal Likelihood

We compute the marginal likelihood by importance sampling by generating S samples from the recognition model and using the following estimator:

$$p(\mathbf{v}) \approx \frac{1}{S}\sum_{s=1}^{S}\frac{p(\mathbf{v}|\mathbf{h}(\boldsymbol{\xi}^{(s)}))p(\boldsymbol{\xi}^{(s)})}{q(\boldsymbol{\xi}^{s})}; \quad \boldsymbol{\xi}^{(s)} \sim q(\boldsymbol{\xi}|\mathbf{v}) \tag{39}$$

E. Missing Data Imputation

Image completion can be approximatively achieved by a simple iterative procedure which consists of (i) initializing the non-observed pixels with random values; (ii) sampling from the recognition distribution given the resulting image; (iii) reconstruct the image given the sample from the recognition model; (iv) iterate the procedure.

We denote the observed and missing entries in an observation as $\mathbf{v}_m, \mathbf{v}_o$, respectively. The imputation procedure can be written formally as a Markov chain with transition kernel $T^q(\mathbf{v}_m \to \mathbf{v}'_m)$ given by

$$T^q(\mathbf{v}_m \to \mathbf{v}'_m) = \int p(\mathbf{v}'|\xi) q(\xi|\mathbf{v}) d\xi, \qquad (40)$$

where $\mathbf{v} = (\mathbf{v}_m, \mathbf{v}_o)$ and $\mathbf{v}' = (\mathbf{v}'_m, \mathbf{v}_o)$.

Provided that the recognition model $q(\xi|\mathbf{v})$ constitutes a good approximation of the true posterior $p(\xi|\mathbf{v})$, (40) can be seen as an approximation of the Kernel

$$T(\mathbf{v}_m \to \mathbf{v}'_m) = \int p(\mathbf{v}'|\xi) p(\xi|\mathbf{v}) d\xi. \qquad (41)$$

The kernel (41) has two important properties: (i) it has as eigen-distribution the model's marginal $p(\mathbf{v}_m|\mathbf{v}_o)$; (ii) $T(\mathbf{v} \to \mathbf{v}') > 0 \forall \mathbf{v}, \mathbf{v}'$. The property (i) can be derived by applying the kernel (41) to the marginal $p(\mathbf{v}_m|\mathbf{v}_o)$ and noting that it is an fixed point. Property (ii) is an immediate consequence of the smoothness of the model.

We apply the fundamental theorem for Markov chains (Neal, 1993, pp. 38) and conclude that given the above properties, a Markov chain generated by (41) is guaranteed to generate samples from the correct marginal $p(\mathbf{v}_m|\mathbf{v}_o)$.

In practice, the stationary distribution of the completed pixels will not be exactly the model's marginal $p(\mathbf{v}_m|\mathbf{v}_o)$, since we can only use the approximated kernel (40). We can nevertheless provide a bound on the L_1 norm of the difference between the resulting stationary marginal and the target marginal $p(\mathbf{v}_m|\mathbf{v}_o)$ by the following proposition.

Proposition E.1 (L_1 bound on marginal error)**.** *If the recognition model $q(\xi|\mathbf{v})$ is such that for all ξ*

$$\exists \varepsilon > 0 \ s.t. \ \int \left| \frac{q(\xi|\mathbf{v})p(\mathbf{v})}{p(\xi)} - p(\mathbf{v}|\xi) \right| d\mathbf{v} \leq \varepsilon \qquad (42)$$

then the marginal $p(\mathbf{v})$ is a weak fixed point of the kernel (40) in the following sense:

$$\int \left| \int \left[T^q(\mathbf{v}_m \to \mathbf{v}'_m) - T(\mathbf{v} \to \mathbf{v}') \right] p(\mathbf{v}) d\mathbf{v} \right| d\mathbf{v}' < \varepsilon. \qquad (43)$$

Proof.

$$\begin{aligned}
&\int \left| \int \left[T^q(\mathbf{v}_m \to \mathbf{v}'_m) - T(\mathbf{v} \to \mathbf{v}') \right] p(\mathbf{v}) d\mathbf{v} \right| d\mathbf{v}' \\
&= \int \left| \int p(\mathbf{v}'|\xi) p(\mathbf{v}) [q(\xi|\mathbf{v}) - p(\xi|\mathbf{v})] d\mathbf{v} d\xi \right| d\mathbf{v}' \\
&= \int \left| \int p(\mathbf{v}'|\xi) p(\mathbf{v}) [q(\xi|\mathbf{v}) - p(\xi|\mathbf{v})] \frac{p(\mathbf{v})}{p(\xi)} \frac{p(\xi)}{p(\mathbf{v})} d\mathbf{v} d\xi \right| d\mathbf{v}' \\
&= \int \left| \int p(\mathbf{v}'|\xi) p(\xi) [q(\xi|\mathbf{v}) \frac{p(\mathbf{v})}{p(\xi)} - p(\mathbf{v}|\xi)] d\mathbf{v} d\xi \right| d\mathbf{v}' \\
&\leq \int \int p(\mathbf{v}'|\xi) p(\xi) \int \left| q(\xi|\mathbf{v}) \frac{p(\mathbf{v})}{p(\xi)} - p(\mathbf{v}|\xi) \right| d\mathbf{v} d\xi d\mathbf{v}' \\
&\leq \varepsilon.
\end{aligned}$$

□

where we apply the condition (42) to obtain the last statement. That is, if the recognition model is sufficiently close to the true posterior to guarantee that (42) holds for some acceptable error ε than (43) guarantees that the fixed point of the Markov chain induced by (40) is not further and ε away from the true marginal with respect to the L_1 norm.

F. Variational Bayes for Deep Directed Models

In the main test we focussed on the variational problem of specifying an posterior on the latent variables only. It is natural to consider the variational Bayes problem in which we specify an approximate posterior for both the latent variables and model parameters.

Following the same construction and considering an Gaussian approximate distribution on the model parameters $\boldsymbol{\theta}^g$, the free energy becomes:

$$\begin{aligned}
\mathcal{F}(\mathbf{V}) = &-\sum_n \overbrace{\mathbb{E}_q[\log p(\mathbf{v}_n|\mathbf{h}(\boldsymbol{\xi}_n))]}^{\text{reconstruction error}} \\
&+ \underbrace{\frac{1}{2}\sum_{n,l} \left[\|\boldsymbol{\mu}_{n,l}\|^2 + \operatorname{Tr} \mathbf{C}_{n,l} - \log |\mathbf{C}_{n,l}| - 1 \right]}_{\text{latent regularization term}} \\
&+ \underbrace{\frac{1}{2}\sum_j \left[\frac{m_j^2}{\kappa} + \frac{\tau_j}{\kappa} + \log \kappa - \log \tau_j - 1 \right]}_{\text{parameter regularization term}}, \qquad (44)
\end{aligned}$$

which now includes an additional term for the cost of using parameters and their regularisation. We must now compute the additional set of gradients with respect to the parameter's mean m_j and variance τ_j are:

$$\nabla_{m_j}\mathcal{F}(\mathbf{v}) = -\mathbb{E}_q\left[\nabla_{\theta_j^g}\log p(\mathbf{v}|\mathbf{h}(\boldsymbol{\xi}))\right] + m_j \tag{45}$$

$$\begin{aligned}\nabla_{\tau_j}\mathcal{F}(\mathbf{v}) &= -\tfrac{1}{2}\mathbb{E}_q\left[\frac{\theta_j - m_j}{\tau_j}\nabla_{\theta_j^g}\log p(\mathbf{v}|\mathbf{h}(\xi))\right] \\ &\quad + \frac{1}{2\kappa} - \frac{1}{2\tau_j}\end{aligned} \tag{46}$$

G. Additional Related Work

There are a number of other aspects of related work, which due to space reasons were not included in the main text, but are worth noting.

General Classes of Latent Gaussian Models. The model class we describe here builds upon other widely-used models with latent Gaussian distributions and provides an inference mechanism for use within this diverse model class. Recognising the latent Gaussian structure shows the connection to models such as generalised linear regression, Gaussian process regression, factor analysis, probabilistic principal components analysis, stochastic volatility models, and other latent Gaussian graphical models (such as log-Gaussian Cox processes and models for covariance selection).

Alternative Inference Approaches. The most common approach for inference in deep directed models has been variational EM. The alternating minimisation often suffers from slow convergence, expensive parameter learning steps, and typically requires the specification of additional local bounds to allows for efficient computation of the required expectations. The wake-sleep algorithm (Dayan, 2000) is a further alternative, but fails to optimise a single consistent objective function and has had limited success in the past. For latent Gaussian models, other alternative inference algorithms have been proposed: the Integrated Nested Laplace Approximation (INLA) has been popular, but it is limited to models with very few hyperparameters (Rue et al., 2009). Alternative variational algorithms such as expectation propagation (EP) (Minka, 2001) can also be used, but are numerically difficult to implement and have performance similar to approaches based on the variational lower bound.