An Introduction to Variational Autoencoders

Other titles in Foundations and Trends® in Machine Learning

Computational Optimal Transport
Gabriel Peyre and Marco Cuturi
ISBN: 978-1-68083-550-2

An Introduction to Deep Reinforcement Learning
Vincent Francois-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare and Joelle Pineau
ISBN: 978-1-68083-538-0

An Introduction to Wishart Matrix Moments
Adrian N. Bishop, Pierre Del Moral and Angele Niclas
ISBN: 978-1-68083-506-9

A Tutorial on Thompson Sampling
Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband and Zheng Wen
ISBN: 978-1-68083-470-3

An Introduction to Variational Autoencoders

Diederik P. Kingma
Google
durk@google.com

Max Welling
University of Amsterdam
Qualcomm
mwelling@qti.qualcomm.com

now
the essence of knowledge
Boston — Delft

Foundations and Trends® in Machine Learning

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

D. P. Kingma and M. Welling. *An Introduction to Variational Autoencoders*. Foundations and Trends® in Machine Learning, vol. 12, no. 4, pp. 307–392, 2019.

ISBN: 978-1-68083-623-3

Foundations and Trends® in Machine Learning
Volume 12, Issue 4, 2019

Editorial Board

Editorial Scope

Topics

Foundations and Trends® in Machine Learning publishes survey and tutorial articles in the following topics:

- Adaptive control and signal processing
- Applications and case studies
- Behavioral, cognitive and neural learning
- Bayesian learning
- Classification and prediction
- Clustering
- Data mining
- Dimensionality reduction
- Evaluation
- Game theoretic learning
- Graphical models
- Independent component analysis
- Inductive logic programming
- Kernel methods
- Markov chain Monte Carlo
- Model choice
- Nonparametric methods
- Online learning
- Optimization
- Reinforcement learning
- Relational learning
- Robustness
- Spectral methods
- Statistical learning theory
- Variational inference
- Visualization

Information for Librarians

Foundations and Trends® in Machine Learning, 2019, Volume 12, 6 issues. ISSN paper version 1935-8237. ISSN online version 1935-8245. Also available as a combined paper and online subscription.

Contents

An Introduction to Variational Autoencoders

Diederik P. Kingma[1] and Max Welling[2,3]

[1] *Google; durk@google.com*
[2] *University of Amsterdam*
[3] *Qualcomm; mwelling@qti.qualcomm.com*

ABSTRACT

Variational autoencoders provide a principled framework for learning deep latent-variable models and corresponding inference models. In this work, we provide an introduction to variational autoencoders and some important extensions.

Diederik P. Kingma and Max Welling (2019), "An Introduction to Variational Autoencoders", Foundations and Trends® in Machine Learning: Vol. 12, No. 4, pp 307–392. DOI: 10.1561/2200000056.

1

Introduction

1.1 Motivation

One major division in machine learning is generative versus discriminative modeling. While in discriminative modeling one aims to learn a predictor given the observations, in generative modeling one aims to solve the more general problem of learning a joint distribution over all the variables. A generative model simulates how the data is generated in the real world. "Modeling" is understood in almost every science as unveiling this generating process by hypothesizing theories and testing these theories through observations. For instance, when meteorologists model the weather they use highly complex partial differential equations to express the underlying physics of the weather. Or when an astronomer models the formation of galaxies s/he encodes in his/her equations of motion the physical laws under which stellar bodies interact. The same is true for biologists, chemists, economists and so on. Modeling in the sciences is in fact almost always generative modeling.

There are many reasons why generative modeling is attractive. First, we can express physical laws and constraints into the generative process while details that we don't know or care about, i.e. nuisance variables, are treated as noise. The resulting models are usually highly intuitive

and interpretable and by testing them against observations we can confirm or reject our theories about how the world works.

Another reason for trying to understand the generative process of data is that it naturally expresses causal relations of the world. Causal relations have the great advantage that they generalize much better to new situations than mere correlations. For instance, once we understand the generative process of an earthquake, we can use that knowledge both in California and in Chile.

To turn a generative model into a discriminator, we need to use Bayes rule. For instance, we have a generative model for an earthquake of type A and another for type B, then seeing which of the two describes the data best we can compute a probability for whether earthquake A or B happened. Applying Bayes rule is however often computationally expensive.

In discriminative methods we directly learn a map in the same direction as we intend to make future predictions in. This is in the opposite direction than the generative model. For instance, one can argue that an image is generated in the world by first identifying the object, then generating the object in 3D and then projecting it onto an pixel grid. A discriminative model takes these pixel values directly as input and maps them to the labels. While generative models can learn efficiently from data, they also tend to make stronger assumptions on the data than their purely discriminative counterparts, often leading to higher asymptotic bias (Banerjee, 2007) when the model is wrong. For this reason, if the model is wrong (and it almost always is to some degree!), if one is solely interested in learning to discriminate, and one is in a regime with a sufficiently large amount of data, then purely discriminative models typically will lead to fewer errors in discriminative tasks. Nevertheless, depending on how much data is around, it may pay off to study the data generating process as a way to guide the training of the discriminator, such as a classifier. For instance, one may have few labeled examples and many more unlabeled examples. In this semi-supervised learning setting, one can use the generative model of the data to improve classification (Kingma *et al.*, 2014; Sønderby *et al.*, 2016a).

Generative modeling can be useful more generally. One can think of it as an auxiliary task. For instance, predicting the immediate future

may help us build useful abstractions of the world that can be used for multiple prediction tasks downstream. This quest for disentangled, semantically meaningful, statistically independent and causal factors of variation in data is generally known as unsupervised representation learning, and the variational autoencoder (VAE) has been extensively employed for that purpose. Alternatively, one may view this as an implicit form of regularization: by forcing the representations to be meaningful for data generation, we bias the inverse of that process, which maps from input to representation, into a certain mould. The auxiliary task of predicting the world is used to better understand the world at an abstract level and thus to better make downstream predictions.

The VAE can be viewed as two coupled, but independently parameterized models: the encoder or recognition model, and the decoder or generative model. These two models support each other. The recognition model delivers to the generative model an approximation to its posterior over latent random variables, which it needs to update its parameters inside an iteration of "expectation maximization" learning. Reversely, the generative model is a scaffolding of sorts for the recognition model to learn meaningful representations of the data, including possibly class-labels. The recognition model is the approximate inverse of the generative model according to Bayes rule.

One advantage of the VAE framework, relative to ordinary Variational Inference (VI), is that the recognition model (also called inference model) is now a (stochastic) function of the input variables. This in contrast to VI where each data-case has a separate variational distribution, which is inefficient for large data-sets. The recognition model uses one set of parameters to model the relation between input and latent variables and as such is called "amortized inference". This recognition model can be arbitrary complex but is still reasonably fast because by construction it can be done using a single feedforward pass from input to latent variables. However the price we pay is that this sampling induces sampling noise in the gradients required for learning. Perhaps the greatest contribution of the VAE framework is the realization that we can counteract this variance by using what is now known as the "reparameterization trick", a simple procedure to reorganize our gradient computation that reduces variance in the gradients.

The VAE is inspired by the Helmholtz Machine (Dayan *et al.*, 1995) which was perhaps the first model that employed a recognition model. However, its wake-sleep algorithm was inefficient and didn't optimize a single objective. The VAE learning rules instead follow from a single approximation to the maximum likelihood objective.

VAEs marry graphical models and deep learning. The generative model is a Bayesian network of the form $p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$, or, if there are multiple stochastic latent layers, a hierarchy such as $p(\mathbf{x}|\mathbf{z}_L)p(\mathbf{z}_L|\mathbf{z}_{L-1})$ $...p(\mathbf{z}_1|\mathbf{z}_0)$. Similarly, the recognition model is also a conditional Bayesian network of the form $q(\mathbf{z}|\mathbf{x})$ or as a hierarchy, such as $q(\mathbf{z}_0|\mathbf{z}_1)...q(\mathbf{z}_L|X)$. But inside each conditional may hide a complex (deep) neural network, e.g. $\mathbf{z}|\mathbf{x} \sim f(\mathbf{x}, \boldsymbol{\epsilon})$, with f a neural network mapping and $\boldsymbol{\epsilon}$ a noise random variable. Its learning algorithm is a mix of classical (amortized, variational) expectation maximization but through the reparameterization trick ends up backpropagating through the many layers of the deep neural networks embedded inside of it.

Since its inception, the VAE framework has been extended in many directions, e.g. to dynamical models (Johnson *et al.*, 2016), models with attention (Gregor *et al.*, 2015), models with multiple levels of stochastic latent variables (Kingma *et al.*, 2016), and many more. It has proven itself as a fertile framework to build new models in. More recently, another generative modeling paradigm has gained significant attention: the generative adversarial network (GAN) (Goodfellow *et al.*, 2014). VAEs and GANs seem to have complementary properties: while GANs can generate images of high subjective perceptual quality, they tend to lack full support over the data (Grover *et al.*, 2018), as opposed to likelihood-based generative models. VAEs, like other likelihood-based models, generate more dispersed samples, but are better density models in terms of the likelihood criterion. As such many hybrid models have been proposed to try to represent the best of both worlds (Dumoulin *et al.*, 2017; Grover *et al.*, 2018; Rosca *et al.*, 2018).

As a community we seem to have embraced the fact that generative models and unsupervised learning play an important role in building intelligent machines. We hope that the VAE provides a useful piece of that puzzle.

1.2 Aim

The framework of *variational autoencoders* (VAEs) (Kingma and Welling, 2014; Rezende *et al.*, 2014) provides a principled method for jointly learning *deep latent-variable models* and corresponding inference models using stochastic gradient descent. The framework has a wide array of applications from generative modeling, semi-supervised learning to representation learning.

This work is meant as an expanded version of our earlier work (Kingma and Welling, 2014), allowing us to explain the topic in finer detail and to discuss a selection of important follow-up work. This is *not* aimed to be a comprehensive review of all related work. We assume that the reader has basic knowledge of algebra, calculus and probability theory.

In this chapter we discuss background material: probabilistic models, directed graphical models, the marriage of directed graphical models with neural networks, learning in fully observed models and deep latent-variable models (DLVMs). In chapter 2 we explain the basics of VAEs. In chapter 3 we explain advanced inference techniques, followed by an explanation of advanced generative models in chapter 4. Please refer to section A.1 for more information on mathematical notation.

1.3 Probabilistic Models and Variational Inference

In the field of machine learning, we are often interested in learning probabilistic models of various natural and artificial phenomena from data. Probabilistic models are mathematical descriptions of such phenomena. They are useful for understanding such phenomena, for prediction of unknowns in the future, and for various forms of assisted or automated decision making. As such, probabilistic models formalize the notion of knowledge and skill, and are central constructs in the field of machine learning and AI.

As probabilistic models contain unknowns and the data rarely paints a complete picture of the unknowns, we typically need to assume some level of uncertainty over aspects of the model. The degree and nature of this uncertainty is specified in terms of (conditional) probability dis-

tributions. Models may consist of both continuous-valued variables and discrete-valued variables. The, in some sense, most complete forms of probabilistic models specify all correlations and higher-order dependencies between the variables in the model, in the form of a joint probability distribution over those variables.

Let's use $\mathbf{x}$ as the vector representing the set of all observed variables whose joint distribution we would like to model. Note that for notational simplicity and to avoid clutter, we use lower case bold (e.g. $\mathbf{x}$) to denote the underlying set of observed random variables, i.e. flattened and concatenated such that the set is represented as a single vector. See section A.1 for more on notation.

We assume the observed variable $\mathbf{x}$ is a random sample from an *unknown underlying process*, whose true (probability) distribution $p^*(\mathbf{x})$ is unknown. We attempt to approximate this underlying process with a chosen model $p_{\boldsymbol{\theta}}(\mathbf{x})$, with parameters $\boldsymbol{\theta}$:

$$\mathbf{x} \sim p_{\boldsymbol{\theta}}(\mathbf{x}) \tag{1.1}$$

Learning is, most commonly, the process of searching for a value of the parameters $\boldsymbol{\theta}$ such that the probability distribution function given by the model, $p_{\boldsymbol{\theta}}(\mathbf{x})$, approximates the true distribution of the data, denoted by $p^*(\mathbf{x})$, such that for any observed $\mathbf{x}$:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) \approx p^*(\mathbf{x}) \tag{1.2}$$

Naturally, we wish $p_{\boldsymbol{\theta}}(\mathbf{x})$ to be sufficiently *flexible* to be able to adapt to the data, such that we have a chance of obtaining a sufficiently accurate model. At the same time, we wish to be able to incorporate knowledge about the distribution of data into the model that is known a priori.

1.3.1 Conditional Models

Often, such as in case of classification or regression problems, we are not interested in learning an unconditional model $p_{\boldsymbol{\theta}}(\mathbf{x})$, but a conditional model $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x})$ that approximates the underlying conditional distribution $p^*(\mathbf{y}|\mathbf{x})$: a distribution over the values of variable $\mathbf{y}$, conditioned on the value of an observed variable $\mathbf{x}$. In this case, $\mathbf{x}$ is often called the *input*

of the model. Like in the unconditional case, a model $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x})$ is chosen, and optimized to be close to the unknown underlying distribution, such that for any $\mathbf{x}$ and $\mathbf{y}$:

$$p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x}) \approx p^*(\mathbf{y}|\mathbf{x}) \tag{1.3}$$

A relatively common and simple example of conditional modeling is image classification, where $\mathbf{x}$ is an image, and $\mathbf{y}$ is the image's class, as labeled by a human, which we wish to predict. In this case, $p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x})$ is typically chosen to be a categorical distribution, whose parameters are computed from $\mathbf{x}$.

Conditional models become more difficult to learn when the predicted variables are very high-dimensional, such as images, video or sound. One example is the reverse of the image classification problem: prediction of a distribution over images, conditioned on the class label. Another example with both high-dimensional input, and high-dimensional output, is time series prediction, such as text or video prediction.

To avoid notational clutter we will often assume unconditional modeling, but one should always keep in mind that the methods introduced in this work are, in almost all cases, equally applicable to conditional models. The data on which the model is conditioned, can be treated as inputs to the model, similar to the parameters of the model, with the obvious difference that one doesn't optimize over their value.

1.4 Parameterizing Conditional Distributions with Neural Networks

Differentiable feed-forward neural networks, from here just called *neural networks*, are a particularly flexible and computationally scalable type of function approximator. Learning of models based on neural networks with multiple 'hidden' layers of artificial neurons is often referred to as *deep learning* (Goodfellow *et al.*, 2016; LeCun *et al.*, 2015). A particularly interesting application is probabilistic models, i.e. the use of neural networks for probability density functions (PDFs) or probability mass functions (PMFs) in probabilistic models. Probabilistic models based on neural networks are computationally scalable since they allow for stochastic gradient-based optimization which, as we will explain,

allows scaling to large models and large datasets. We will denote a deep neural network as a vector function: NeuralNet(.).

At the time of writing, deep learning has been shown to work well for a large variety of classification and regression problems, as summarized in (LeCun *et al.*, 2015; Goodfellow *et al.*, 2016). In case of neural-network based image classification LeCun *et al.*, 1998, for example, neural networks parameterize a categorical distribution $p_{\boldsymbol{\theta}}(y|\mathbf{x})$ over a class label y, conditioned on an image $\mathbf{x}$.

$$\mathbf{p} = \text{NeuralNet}(\mathbf{x}) \tag{1.4}$$

$$p_{\boldsymbol{\theta}}(y|\mathbf{x}) = \text{Categorical}(y; \mathbf{p}) \tag{1.5}$$

where the last operation of NeuralNet(.) is typically a softmax() function such that $\sum_i p_i = 1$.

1.5 Directed Graphical Models and Neural Networks

We work with *directed* probabilistic models, also called directed *probabilistic graphical models* (PGMs), or *Bayesian networks*. Directed graphical models are a type of probabilistic models where all the variables are topologically organized into a directed acyclic graph. The joint distribution over the variables of such models factorizes as a product of prior and conditional distributions:

$$p_{\boldsymbol{\theta}}(\mathbf{x}_1, ..., \mathbf{x}_M) = \prod_{j=1}^{M} p_{\boldsymbol{\theta}}(\mathbf{x}_j|Pa(\mathbf{x}_j)) \tag{1.6}$$

where $Pa(\mathbf{x}_j)$ is the set of parent variables of node j in the directed graph. For non-root-nodes, we condition on the parents. For root nodes, the set of parents is the empty set, such that the distribution is unconditional.

Traditionally, each conditional probability distribution $p_{\boldsymbol{\theta}}(\mathbf{x}_j|Pa(\mathbf{x}_j))$ is parameterized as a lookup table or a linear model (Koller and Friedman, 2009). As we explained above, a more flexible way to parameterize such conditional distributions is with neural networks. In this case, neural networks take as input the parents of a variable in a directed

graph, and produce the distributional parameters $\boldsymbol{\eta}$ over that variable:

$$\boldsymbol{\eta} = \text{NeuralNet}(Pa(\mathbf{x})) \tag{1.7}$$

$$p_{\boldsymbol{\theta}}(\mathbf{x}|Pa(\mathbf{x})) = p_{\boldsymbol{\theta}}(\mathbf{x}|\boldsymbol{\eta}) \tag{1.8}$$

We will now discuss how to learn the parameters of such models, if all the variables are observed in the data.

1.6 Learning in Fully Observed Models with Neural Nets

If all variables in the directed graphical model are observed in the data, then we can compute and differentiate the log-probability of the data under the model, leading to relatively straightforward optimization.

1.6.1 Dataset

We often collect a dataset $\mathcal{D}$ consisting of $N \geq 1$ datapoints:

$$\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}\} \equiv \{\mathbf{x}^{(i)}\}_{i=1}^{N} \equiv \mathbf{x}^{(1:N)} \tag{1.9}$$

The datapoints are assumed to be independent samples from an unchanging underlying distribution. In other words, the dataset is assumed to consist of distinct, independent measurements from the same (unchanging) system. In this case, the observations $\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^{N}$ are said to be *i.i.d.*, for *independently and identically distributed*. Under the i.i.d. assumption, the probability of the datapoints given the parameters factorizes as a product of individual datapoint probabilities. The log-probability assigned to the data by the model is therefore given by:

$$\log p_{\boldsymbol{\theta}}(\mathcal{D}) = \sum_{\mathbf{x} \in \mathcal{D}} \log p_{\boldsymbol{\theta}}(\mathbf{x}) \tag{1.10}$$

1.6.2 Maximum Likelihood and Minibatch SGD

The most common criterion for probabilistic models is *maximum log-likelihood* (ML). As we will explain, maximization of the log-likelihood criterion is equivalent to minimization of a Kullback Leibler divergence between the data and model distributions.

Under the ML criterion, we attempt to find the parameters $\boldsymbol{\theta}$ that maximize the sum, or equivalently the average, of the log-probabilities

assigned to the data by the model. With i.i.d. dataset $\mathcal{D}$ of size $N_{\mathcal{D}}$, the maximum likelihood objective is to maximize the log-probability given by equation (1.10).

Using calculus' chain rule and automatic differentiation tools, we can efficiently compute gradients of this objective, i.e. the first derivatives of the objective w.r.t. its parameters $\boldsymbol{\theta}$. We can use such gradients to iteratively hill-climb to a local optimum of the ML objective. If we compute such gradients using all datapoints, $\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathcal{D})$, then this is known as *batch* gradient descent. Computation of this derivative is, however, an expensive operation for large dataset size $N_{\mathcal{D}}$, since it scales linearly with $N_{\mathcal{D}}$.

A more efficient method for optimization is *stochastic gradient descent* (SGD) (section A.3), which uses randomly drawn minibatches of data $\mathcal{M} \subset \mathcal{D}$ of size $N_{\mathcal{M}}$. With such minibatches we can form an unbiased estimator of the ML criterion:

$$\frac{1}{N_{\mathcal{D}}} \log p_{\boldsymbol{\theta}}(\mathcal{D}) \simeq \frac{1}{N_{\mathcal{M}}} \log p_{\boldsymbol{\theta}}(\mathcal{M}) = \frac{1}{N_{\mathcal{M}}} \sum_{\mathbf{x} \in \mathcal{M}} \log p_{\boldsymbol{\theta}}(\mathbf{x}) \tag{1.11}$$

The $\simeq$ symbol means that one of the two sides is an *unbiased estimator* of the other side. So one side (in this case the right-hand side) is a random variable due to some noise source, and the two sides are equal when averaged over the noise distribution. The noise source, in this case, is the randomly drawn minibatch of data $\mathcal{M}$. The unbiased estimator $\log p_{\boldsymbol{\theta}}(\mathcal{M})$ is differentiable, yielding the unbiased stochastic gradients:

$$\frac{1}{N_{\mathcal{D}}} \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathcal{D}) \simeq \frac{1}{N_{\mathcal{M}}} \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathcal{M}) = \frac{1}{N_{\mathcal{M}}} \sum_{\mathbf{x} \in \mathcal{M}} \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{x}) \tag{1.12}$$

These gradients can be plugged into stochastic gradient-based optimizers; see section A.3 for further discussion. In a nutshell, we can optimize the objective function by repeatedly taking small steps in the direction of the stochastic gradient.

1.6.3 Bayesian inference

From a Bayesian perspective, we can improve upon ML through *maximum a posteriori* (MAP) estimation (section section A.2.1), or, going

even further, inference of a full approximate posterior distribution over the parameters (see section A.1.4).

1.7 Learning and Inference in Deep Latent Variable Models

1.7.1 Latent Variables

We can extend fully-observed directed models, discussed in the previous section, into directed models with *latent variables*. Latent variables are variables that are part of the model, but which we don't observe, and are therefore not part of the dataset. We typically use $\mathbf{z}$ to denote such latent variables. In case of unconditional modeling of observed variable $\mathbf{x}$, the directed graphical model would then represent a joint distribution $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ over both the observed variables $\mathbf{x}$ and the latent variables $\mathbf{z}$. The marginal distribution over the observed variables $p_{\boldsymbol{\theta}}(\mathbf{x})$, is given by:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\, d\mathbf{z} \tag{1.13}$$

This is also called the (single datapoint) *marginal likelihood* or the *model evidence*, when taken as a function of $\boldsymbol{\theta}$.

Such an implicit distribution over $\mathbf{x}$ can be quite flexible. If $\mathbf{z}$ is discrete and $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ is a Gaussian distribution, then $p_{\boldsymbol{\theta}}(\mathbf{x})$ is a mixture-of-Gaussians distribution. For continuous $\mathbf{z}$, $p_{\boldsymbol{\theta}}(\mathbf{x})$ can be seen as an infinite mixture, which are potentially more powerful than discrete mixtures. Such marginal distributions are also called compound probability distributions.

1.7.2 Deep Latent Variable Models

We use the term *deep latent variable model* (DLVM) to denote a latent variable model $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ whose distributions are parameterized by neural networks. Such a model can be conditioned on some context, like $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}|\mathbf{y})$. One important advantage of DLVMs, is that even when each factor (prior or conditional distribution) in the directed model is relatively simple (such as conditional Gaussian), the marginal distribution $p_{\boldsymbol{\theta}}(\mathbf{x})$ can be very complex, i.e. contain almost arbitrary dependen-

cies. This expressivity makes deep latent-variable models attractive for approximating complicated underlying distributions $p^*(\mathbf{x})$.

Perhaps the simplest, and most common, DLVM is one that is specified as factorization with the following structure:

$$p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) = p_{\boldsymbol{\theta}}(\mathbf{z}) p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \tag{1.14}$$

where $p_{\boldsymbol{\theta}}(\mathbf{z})$ and/or $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ are specified. The distribution $p(\mathbf{z})$ is often called the *prior distribution* over $\mathbf{z}$, since it is not conditioned on any observations.

1.7.3 Example DLVM for multivariate Bernoulli data

A simple example DLVM, used in (Kingma and Welling, 2014) for binary data $\mathbf{x}$, is with a spherical Gaussian latent space, and a factorized Bernoulli observation model:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, \mathbf{I}) \tag{1.15}$$

$$\mathbf{p} = \text{DecoderNeuralNet}_{\boldsymbol{\theta}}(\mathbf{z}) \tag{1.16}$$

$$\log p(\mathbf{x}|\mathbf{z}) = \sum_{j=1}^{D} \log p(x_j|\mathbf{z}) = \sum_{j=1}^{D} \log \text{Bernoulli}(x_j; p_j) \tag{1.17}$$

$$= \sum_{j=1}^{D} x_j \log p_j + (1 - x_j) \log(1 - p_j) \tag{1.18}$$

where $\forall p_j \in \mathbf{p} : 0 \leq p_j \leq 1$ (e.g. implemented through a sigmoid nonlinearity as the last layer of the $\text{DecoderNeuralNet}_{\boldsymbol{\theta}}(.)$), where D is the dimensionality of $\mathbf{x}$, and $\text{Bernoulli}(.; p)$ is the probability mass function (PMF) of the Bernoulli distribution.

1.8 Intractabilities

The main difficulty of maximum likelihood learning in DLVMs is that the marginal probability of data under the model is typically intractable. This is due to the integral in equation (1.13) for computing the marginal likelihood (or model evidence), $p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})\, d\mathbf{z}$, not having an analytic solution or efficient estimator. Due to this intractability, we

cannot differentiate it w.r.t. its parameters and optimize it, as we can with fully observed models.

The intractability of $p_{\boldsymbol{\theta}}(\mathbf{x})$, is related to the intractability of the posterior distribution $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$. Note that the joint distribution $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ is efficient to compute, and that the densities are related through the basic identity:

$$p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})}{p_{\boldsymbol{\theta}}(\mathbf{x})} \tag{1.19}$$

Since $p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})$ *is* tractable to compute, a tractable marginal likelihood $p_{\boldsymbol{\theta}}(\mathbf{x})$ leads to a tractable posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, and vice versa. Both are intractable in DLVMs.

Approximate inference techniques (see also section A.2) allow us to approximate the posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$ and the marginal likelihood $p_{\boldsymbol{\theta}}(\mathbf{x})$ in DLVMs. Traditional inference methods are relatively expensive. Such methods, for example, often require a per-datapoint optimization loop, or yield bad posterior approximations. We would like to avoid such expensive procedures.

Likewise, the posterior over the parameters of (directed models parameterized with) neural networks, $p(\boldsymbol{\theta}|\mathcal{D})$, is generally intractable to compute exactly, and requires approximate inference techniques.

Acknowledgements

We are grateful for the help of Tim Salimans, Alec Radford, Rif A. Saurous and others who have given us valuable feedback at various stages of writing.

Appendices

A

Appendix

A.1 Notation and definitions

A.1.1 Notation

Example(s)	Description
$\mathbf{x}, \mathbf{y}\, \mathbf{z}$	With characters in bold we typically denote random *vectors*. We also use this notation for collections of random variables variables.
x, y, z	With characters in italic we typically denote random *scalars*, i.e. single real-valued numbers.
$\mathbf{X}, \mathbf{Y}, \mathbf{Z}$	With bold and capitalized letters we typically denote random *matrices*.
$Pa(\mathbf{z})$	The parents of random variable $\mathbf{z}$ in a directed graph.
$\text{diag}(\mathbf{x})$	Diagonal matrix, with the values of vector $\mathbf{x}$ on the diagonal.

$\mathbf{x} \odot \mathbf{y}$	Element-wise multiplication of two vectors. The resulting vector is $(x_1 y_1, ..., x_K y_K)^T$.
θ	Parameters of a (generative) model are typically denoted with the Greek lowercase letter θ (theta).
$\boldsymbol{\phi}$	Variational parameters are typically denoted with the bold Greek letter $\boldsymbol{\phi}$ (phi).
$p(\mathbf{x}), p(\mathbf{z})$	Probability density functions (PDFs) and probability mass functions (PMFs), also simply called *distributions*, are denoted by $p(.)$, $q(.)$ or $r(.)$.
$p(\mathbf{x}, \mathbf{y}, \mathbf{z})$	Joint distributions are denoted by $p(.,.)$
$p(\mathbf{x}\|\mathbf{z})$	Conditional distributions are denoted by $p(.\|.)$
$p(.;\theta), p_\theta(\mathbf{x})$	The parameters of a distribution are denoted with $p(.;\theta)$ or equivalently with subscript $p_\theta(.)$.
$p(\mathbf{x} = \mathbf{a})$, $p(\mathbf{x} \leq \mathbf{a})$	We may use an (in-)equality sign within a probability distribution to distinguish between function arguments and value at which to evaluate. So $p(\mathbf{x} = \mathbf{a})$ denotes a PDF or PMF over variable $\mathbf{x}$ evaluated at the value of variable $\mathbf{a}$. Likewise, $p(\mathbf{x} \leq \mathbf{a})$ denotes a CDF evaluated at the value of $\mathbf{a}$.
$p(.), q(.)$	We use different letters to refer to different probabilistic models, such as $p(.)$ or $q(.)$. Conversely, we use the *same* letter across different marginals/conditionals to indicate they relate to the same probabilistic model.

A.1.2 Definitions

Term	**Description**

Probability density function (PDF)	A function that assigns a probability *density* to each possible value of given *continuous* random variables.
Cumulative distribution function (CDF)	A function that assigns a cumulative probability density to each possible value of given univariate *continuous* random variables.
Probability mass function (PMF)	A function that assigns a probability *mass* to given *discrete* random variable.

A.1.3 Distributions

We overload the notation of distributions (e.g. $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$) with two meanings: (1) a distribution from which we can sample, and (2) the probability density function (PDF) of that distribution.

Term	**Description**
Categorical$(x; \mathbf{p})$	Categorical distribution, with parameter $\mathbf{p}$ such that $\sum_i p_i = 1$.
Bernoulli$(\mathbf{x}; \mathbf{p})$	Multivariate distribution of independent Bernoulli. Bernoulli$(\mathbf{x}; \mathbf{p}) = \prod_i \text{Bernoulli}(x_i; p_i)$ with $\forall i : 0 \leq p_i \leq 1$.
Normal$(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Multivariate Normal distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$.

Chain rule of probability

$$p(\mathbf{a}, \mathbf{b}) = p(\mathbf{a})p(\mathbf{b}|\mathbf{a}) \tag{A.1}$$

Bayes' Rule

$$p(\mathbf{a}|\mathbf{b}) = p(\mathbf{b}|\mathbf{a})p(\mathbf{a})/p(\mathbf{b}) \tag{A.2}$$

A.1.4 Bayesian Inference

Let $p(\theta)$ be a chosen marginal distribution over its parameters θ, called a *prior distribution*. Let $\mathcal{D}$ be observed data, $p(\mathcal{D}|\theta) \equiv p_{\boldsymbol{\theta}}(\mathcal{D})$ be the probability assigned to the data under the model with parameters θ. Recall the chain rule in probability:

$$p(\theta, \mathcal{D}) = p(\theta|\mathcal{D})p(\mathcal{D}) = p(\theta)p(\mathcal{D}|\theta)$$

Simply re-arranging terms above, the posterior distribution over the parameters θ, taking into account the data $\mathcal{D}$, is:

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} \propto p(\mathcal{D}|\theta)p(\theta) \tag{A.3}$$

where the proportionality ($\propto$) holds since $p(\mathcal{D})$ is a constant that is not dependent on parameters θ. The formula above is known as *Bayes' rule*, a fundamental formula in machine learning and statistics, and is of special importance to this work.

A principal application of Bayes' rule is that it allows us to make predictions about future data $\mathbf{x}'$, that are optimal as long as the prior $p(\theta)$ and model class $p_{\boldsymbol{\theta}}(\mathbf{x})$ are correct:

$$p(\mathbf{x} = \mathbf{x}'|\mathcal{D}) = \int p_{\boldsymbol{\theta}}(\mathbf{x} = \mathbf{x}')p(\theta|\mathcal{D})d\theta$$

A.2 Alternative methods for learning in DLVMs

A.2.1 Maximum A Posteriori

From a Bayesian perspective, we can improve upon the maximum likelihood objective through *maximum a posteriori* (MAP) estimation, which maximizes the log-posterior w.r.t. θ. With i.i.d. data $\mathcal{D}$, this is:

$$L^{MAP}(\theta) = \log p(\theta|\mathcal{D}) \tag{A.4}$$

$$= \log p(\theta) + L^{ML}(\theta) + \text{constant} \tag{A.5}$$

The prior $p(\theta)$ in equation (A.5) has diminishing effect for increasingly large N. For this reason, in case of optimization with large datasets, we often choose to simply use the maximum likelihood criterion by omitting the prior from the objective, which is numerically equivalent to setting $p(\theta) = \text{constant}$.

A.2.2 Variational EM with local variational parameters

Expectation Maximization (EM) is a general strategy for learning parameters in partially observed models (Dempster *et al.*, 1977). See section A.2.3 for a discussion of EM using MCMC. The method can be explained as coordinate ascent on the ELBO (Neal and Hinton, 1998). In case of of i.i.d. data, traditional variational EM methods estimate **local variational parameters** $\boldsymbol{\phi}^{(i)}$, i.e. a separate set of variational parameters per datapoint i in the dataset. In contrast, VAEs employ a strategy with **global variational parameters**.

EM starts out with some (random) initial choice of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}^{(1:N)}$. It then iteratively applies updates:

$$\forall i = 1, ..., N: \quad \boldsymbol{\phi}^{(i)} \leftarrow \operatorname*{argmax}_{\boldsymbol{\phi}} \mathcal{L}(\mathbf{x}^{(i)}; \boldsymbol{\theta}, \boldsymbol{\phi}) \qquad \text{(E-step)} \qquad \text{(A.6)}$$

$$\boldsymbol{\theta} \leftarrow \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{x}^{(i)}; \boldsymbol{\theta}, \boldsymbol{\phi}) \qquad \text{(M-step)} \qquad \text{(A.7)}$$

until convergence. Why does this work? Note that at the E-step:

$$\operatorname*{argmax}_{\boldsymbol{\phi}} \mathcal{L}(\mathbf{x}; \boldsymbol{\theta}, \boldsymbol{\phi}) \qquad \text{(A.8)}$$

$$= \operatorname*{argmax}_{\boldsymbol{\phi}} \left[\log p_{\boldsymbol{\theta}}(\mathbf{x}) - D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}))\right] \qquad \text{(A.9)}$$

$$= \operatorname*{argmin}_{\boldsymbol{\phi}} D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})) \qquad \text{(A.10)}$$

so the E-step, sensibly, minimizes the KL divergence of $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ from the true posterior.

Secondly, note that if $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ equals $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x})$, the ELBO equals the marginal likelihood, but that for any choice of $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$, the M-step optimizes a bound on the marginal likelihood. The tightness of this bound is defined by $D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}))$.

A.2.3 MCMC-EM

Another Bayesian approach towards optimizing the likelihood $p_{\boldsymbol{\theta}}(\mathbf{x})$ with DLVMs is Expectation Maximization (EM) with Markov Chain Monte Carlo (MCMC). In case of MCMC, the posterior is approximated by a mixture of a set of approximately i.i.d. samples from the posterior, acquired by running a Markov chain. Note that posterior gradients in DLVMs are relatively affordable to compute by differentiating the log-joint distribution w.r.t. $\mathbf{z}$:

$$\begin{aligned}
\nabla_{\mathbf{z}} \log p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}) &= \nabla_{\mathbf{z}} \log[p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z})/p_{\boldsymbol{\theta}}(\mathbf{x})] && \text{(A.11)} \\
&= \nabla_{\mathbf{z}}[\log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) - \log p_{\boldsymbol{\theta}}(\mathbf{x})] && \text{(A.12)} \\
&= \nabla_{\mathbf{z}} \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) - \nabla_{\mathbf{z}} \log p_{\boldsymbol{\theta}}(\mathbf{x}) && \text{(A.13)} \\
&= \nabla_{\mathbf{z}} \log p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}) && \text{(A.14)}
\end{aligned}$$

One version of MCMC which uses such posterior for relatively fast convergence, is Hamiltonian MCMC (Neal, 2011). A disadvantage of this approach is the requirement for running an independent MCMC chain per datapoint.

A.3 Stochastic Gradient Descent

We work with directed models where the objective per datapoint is scalar, and due to the differentiability of neural networks that compose them, the objective is differentiable w.r.t. its parameters θ. Due to the remarkable efficiency of reverse-mode automatic differentiation (also known as the backpropagation algorithm (Rumelhart *et al.*, 1988)), the value and gradient (i.e. the vector of partial derivatives) of differentiable scalar objectives can be computed with equal time complexity. In SGD, we iteratively update parameters θ:

$$\theta_{t+1} \leftarrow \theta_t + \alpha_t \cdot \nabla_\theta \tilde{L}(\theta, \xi) \tag{A.15}$$

where α_t is a learning rate or preconditioner, and $\tilde{L}(\theta, \xi)$ is an unbiased estimate of the objective $L(\theta)$, i.e. $\mathbb{E}_{\xi \sim p(\xi)}\left[\tilde{L}(\theta, \xi)\right] = L(\theta)$. The random variable ξ could e.g. be a datapoint index, uniformly sampled from $\{1, ..., N\}$, but can also include different types of noise such posterior sampling noise in VAEs. In experiments, we have typically used the

Adam and Adamax optimization methods for choosing α_t (Kingma and Ba, 2015); these methods are invariant to constant rescaling of the objective, and invariant to constant re-scalings of the individual gradients. As a result, $\tilde{L}(\theta, \xi)$ only needs to be unbiased up to proportionality. We iteratively apply eq. (A.15) until a stopping criterion is met. A simple but effective criterion is to stop optimization as soon as the probability of a holdout set of data starts decreasing; this criterion is called *early stopping*.

References

Banerjee, A. 2007. "An analysis of logistic models: Exponential family connections and online performance". In: *Proceedings of the 2007 SIAM International Conference on Data Mining*. SIAM. 204–215.

Bayer, J. and C. Osendorfer. 2014. "Learning stochastic recurrent networks". In: *NIPS 2014 Workshop on Advances in Variational Inference*.

Bengio, Y., A. Courville, and P. Vincent. 2013. *Representation Learning: A Review and New Perspectives*. IEEE.

Bengio, Y., E. Laufer, G. Alain, and J. Yosinski. 2014. "Deep generative stochastic networks trainable by backprop". In: *International Conference on Machine Learning*. 226–234.

Berg, R. v. d., L. Hasenclever, J. M. Tomczak, and M. Welling. 2017. "Sylvester Normalizing Flows for Variational Inference". *Conference on Uncertainty in Artificial Intelligence*.

Blei, D. M., M. I. Jordan, and J. W. Paisley. 2012. "Variational Bayesian inference with stochastic search". In: *International Conference on Machine Learning*. 1367–1374.

Blundell, C., J. Cornebise, K. Kavukcuoglu, and D. Wierstra. 2015. "Weight Uncertainty in Neural Networks". In: *International Conference on Machine Learning*. 1613–1622.

Bornschein, J., S. Shabanian, A. Fischer, and Y. Bengio. 2016. "Bidirectional Helmholtz machines". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning*. 2511–2519.

Bourlard, H. and Y. Kamp. 1988. "Auto-association by multilayer perceptrons and singular value decomposition". *Biological Cybernetics*. 59(4-5): 291–294.

Bowman, S. R., L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. 2015. "Generating sentences from a continuous space". *arXiv preprint arXiv:1511.06349*.

Brock, A., T. Lim, J. M. Ritchie, and N. J. Weston. 2017. "Neural photo editing with introspective adversarial networks". In: *International Conference on Learning Representations*.

Burda, Y., R. Grosse, and R. Salakhutdinov. 2015. "Importance weighted autoencoders". *arXiv preprint arXiv:1509.00519*.

Chen, R. T., X. Li, R. Grosse, and D. Duvenaud. 2018. "Isolating sources of disentanglement in VAEs". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. Curran Associates Inc. 2615–2625.

Chen, X., D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel. 2017. "Variational lossy autoencoder". *International Conference on Learning Representations*.

Chung, J., K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. 2015. "A recurrent latent variable model for sequential data". In: *Advances in neural information processing systems*. 2980–2988.

Cremer, C., Q. Morris, and D. Duvenaud. 2017. "Re-interpreting importance weighted autoencoders". *International Conference on Learning Representations*.

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

Deco, G. and W. Brauer. 1995. "Higher order statistical decorrelation without information loss". *Advances in Neural Information Processing Systems*: 247–254.

Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum likelihood from incomplete data via the EM algorithm". *Journal of the Royal Statistical Society. Series B (Methodological)*: 1–38.

Deshpande, A., J. Lu, M.-C. Yeh, M. Jin Chong, and D. Forsyth. 2017. "Learning diverse image colorization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6837–6845.

Dinh, L., D. Krueger, and Y. Bengio. 2014. "NICE: Non-linear independent components estimation". *arXiv preprint arXiv:1410.8516*.

Dinh, L., J. Sohl-Dickstein, and S. Bengio. 2016. "Density estimation using Real NVP". *arXiv preprint arXiv:1605.08803*.

Dosovitskiy, A., J. Tobias Springenberg, and T. Brox. 2015. "Learning to generate chairs with convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1538–1546.

Dumoulin, V., I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. 2017. "Adversarially learned inference". *International Conference on Learning Representations*.

Edwards, H. and A. Storkey. 2017. "Towards a neural statistician". *International Conference on Learning Representations*.

Eslami, S. A., N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, *et al.* 2016. "Attend, infer, repeat: Fast scene understanding with generative models". In: *Advances In Neural Information Processing Systems*. 3225–3233.

Fan, K., Z. Wang, J. Beck, J. Kwok, and K. A. Heller. 2015. "Fast second order stochastic backpropagation for variational inference". In: *Advances in Neural Information Processing Systems*. 1387–1395.

Fortunato, M., C. Blundell, and O. Vinyals. 2017. "Bayesian recurrent neural networks". *arXiv preprint arXiv:1704.02798*.

Fraccaro, M., S. K. Sønderby, U. Paquet, and O. Winther. 2016. "Sequential neural models with stochastic layers". In: *Advances in Neural Information Processing Systems*. 2199–2207.

Fu, M. C. 2006. "Gradient estimation". *Handbooks in Operations Research and Management Science*. 13: 575–616.

Gal, Y. and Z. Ghahramani. 2016. "A theoretically grounded application of dropout in recurrent neural networks". In: *Advances in neural information processing systems*. 1019–1027.

Germain, M., K. Gregor, I. Murray, and H. Larochelle. 2015. "Made: Masked autoencoder for distribution estimation". In: *International Conference on Machine Learning*. 881–889.

Gershman, S. and N. Goodman. 2014. "Amortized inference in probabilistic reasoning." In: *CogSci*.

Glasserman, P. 2013. *Monte Carlo methods in financial engineering*. Vol. 53. Springer Science & Business Media.

Glynn, P. W. 1990. "Likelihood ratio gradient estimation for stochastic systems". *Communications of the ACM*. 33(10): 75–84.

Gómez-Bombarelli, R., J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik. 2018. "Automatic chemical design using a data-driven continuous representation of molecules". *ACS central science*. 4(2): 268–276.

Goodfellow, I., Y. Bengio, and A. Courville. 2016. *Deep learning*. MIT press.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. "Generative adversarial nets". In: *Advances in Neural Information Processing Systems*. 2672–2680.

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

Gregor, K., F. Besse, D. J. Rezende, I. Danihelka, and D. Wierstra. 2016. "Towards conceptual compression". In: *Advances In Neural Information Processing Systems*. 3549–3557.

Gregor, K., I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. 2015. "DRAW: A Recurrent Neural Network For Image Generation". In: *International Conference on Machine Learning*. 1462–1471.

Gregor, K., I. Danihelka, A. Mnih, C. Blundell, and D. Wierstra. 2014. "Deep AutoRegressive Networks". In: *International Conference on Machine Learning*. 1242–1250.

Grover, A., M. Dhar, and S. Ermon. 2018. "Flow-GAN: Combining maximum likelihood and adversarial learning in generative models". In: *AAAI Conference on Artificial Intelligence*.

Gu, S., S. Levine, I. Sutskever, and A. Mnih. 2015. "MuProp: Unbiased backpropagation for stochastic neural networks". *arXiv preprint arXiv:1511.05176*.

Gulrajani, I., K. Kumar, F. Ahmed, A. A. Taiga, F. Visin, D. Vazquez, and A. Courville. 2017. "PixelVAE: A latent variable model for natural images". *International Conference on Learning Representations*.

He, K., X. Zhang, S. Ren, and J. Sun. 2015. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE International Conference on Computer Vision*. 1026–1034.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

Heess, N., G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa. 2015. "Learning continuous control policies by stochastic value gradients". In: *Advances in Neural Information Processing Systems*. 2944–2952.

Hernández-Lobato, J. M., Y. Li, M. Rowland, D. Hernández-Lobato, T. Bui, and R. E. Turner. 2016. "Black-box α-divergence minimization".

Higgins, I., L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. 2017. "beta-vae: Learning basic visual concepts with a constrained variational framework". *International Conference on Learning Representations*.

Hinton, G. E., P. Dayan, B. J. Frey, and R. M. Neal. 1995. "The "Wake-Sleep" algorithm for unsupervised neural networks". *Science*: 1158–1158.

Hochreiter, S. and J. Schmidhuber. 1997. "Long Short-Term Memory". *Neural Computation*. 9(8): 1735–1780.

Hoffman, M. D., D. M. Blei, C. Wang, and J. Paisley. 2013. "Stochastic variational inference". *The Journal of Machine Learning Research*. 14(1): 1303–1347.

Hoffman, M. D. and M. J. Johnson. 2016. "Elbo surgery: yet another way to carve up the variational evidence lower bound". In: *Workshop in Advances in Approximate Bayesian Inference, NIPS*.

Houthooft, R., X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. 2016. "Vime: Variational information maximizing exploration". In: *Advances in Neural Information Processing Systems*. 1109–1117.

Hsu, C.-C., H.-T. Hwang, Y.-C. Wu, Y. Tsao, and H.-M. Wang. 2016. "Voice conversion from non-parallel corpora using variational auto-encoder". In: *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2016 Asia-Pacific*. IEEE. 1–6.

Hsu, C.-C., H.-T. Hwang, Y.-C. Wu, Y. Tsao, and H.-M. Wang. 2017. "Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks". *arXiv preprint arXiv:1704.00849*.

Hu, Z., Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing. 2017. "Controllable text generation". *arXiv preprint arXiv:1703.00955*.

Huang, C.-W., D. Krueger, A. Lacoste, and A. Courville. 2018. "Neural Autoregressive Flows". In: *International Conference on Machine Learning*. 2083–2092.

Jang, E., S. Gu, and B. Poole. 2017. "Categorical Reparameterization with Gumbel-Softmax". *International Conference on Learning Representations*.

Johnson, M., D. K. Duvenaud, A. Wiltschko, R. P. Adams, and S. R. Datta. 2016. "Composing graphical models with neural networks for structured representations and fast inference". In: *Advances in Neural Information Processing Systems*. 2946–2954.

Jozefowicz, R., W. Zaremba, and I. Sutskever. 2015. "An empirical exploration of recurrent network architectures". In: *International Conference on Machine Learning*. 2342–2350.

Karl, M., M. Soelch, J. Bayer, and P. van der Smagt. 2017. "Deep variational bayes filters: Unsupervised learning of state space models from raw data". *International Conference on Learning Representations*.

Kavukcuoglu, K., M. Ranzato, and Y. LeCun. 2008. "Fast inference in sparse coding algorithms with applications to object recognition". *Tech. rep.* No. CBLL-TR-2008-12-01. Computational and Biological Learning Lab, Courant Institute, NYU.

Kingma, D. P., S. Mohamed, D. J. Rezende, and M. Welling. 2014. "Semi-supervised learning with deep generative models". In: *Advances in Neural Information Processing Systems*. 3581–3589.

Kingma, D. P., T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling. 2016. "Improved variational inference with inverse autoregressive flow". In: *Advances in Neural Information Processing Systems*. 4743–4751.

Kingma, D. P., T. Salimans, and M. Welling. 2015. "Variational dropout and the local reparameterization trick". In: *Advances in Neural Information Processing Systems*. 2575–2583.

Kingma, D. P. and M. Welling. 2014. "Auto-Encoding Variational Bayes". *International Conference on Learning Representations*.

Kingma, D. and J. Ba. 2015. "Adam: A Method for Stochastic Optimization". *International Conference on Learning Representations*.

Kipf, T. N. and M. Welling. 2016. "Variational graph auto-encoders". *arXiv preprint arXiv:1611.07308*.

Kleijnen, J. P. and R. Y. Rubinstein. 1996. "Optimization and sensitivity analysis of computer simulation models by the score function method". *European Journal of Operational Research*. 88(3): 413–427.

Koller, D. and N. Friedman. 2009. *Probabilistic graphical models: Principles and techniques*. MIT press.

Krishnan, R. G., U. Shalit, and D. Sontag. 2017. "Structured Inference Networks for Nonlinear State Space Models." In: *AAAI*. 2101–2109.

Kucukelbir, A., D. Tran, R. Ranganath, A. Gelman, and D. M. Blei. 2017. "Automatic differentiation variational inference". *The Journal of Machine Learning Research*. 18(1): 430–474.

Kulkarni, T. D., W. F. Whitney, P. Kohli, and J. Tenenbaum. 2015. "Deep convolutional inverse graphics network". In: *Advances in Neural Information Processing Systems*. 2539–2547.

Kusner, M. J., B. Paige, and J. M. Hernández-Lobato. 2017. "Grammar variational autoencoder". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 1945–1954.

Larsen, A. B. L., S. K. Sønderby, H. Larochelle, and O. Winther. 2016. "Autoencoding beyond pixels using a learned similarity metric". In: *International Conference on Machine Learning*. 1558–1566.

Lázaro-Gredilla, M. 2014. "Doubly stochastic variational Bayes for non-conjugate inference". In: International Conference on Machine Learning.

LeCun, Y., Y. Bengio, and G. Hinton. 2015. "Deep Learning". *Nature*. 521(7553): 436–444.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. "Gradient-based learning applied to document recognition". *Proceedings of the IEEE*. 86(11): 2278–2324.

Li, Y. and R. E. Turner. 2016. "Rényi divergence variational inference". In: *Advances in Neural Information Processing Systems*. 1073–1081.

Li, Y., K. Swersky, and R. S. Zemel. 2015. "Generative moment matching networks". In: *International Conference on Machine Learning*. 1718–1727.

Linsker, R. 1989. *An Application of the Principle of Maximum Information Preservation to Linear Systems*. Morgan Kaufmann Publishers Inc.

Louizos, C., K. Swersky, Y. Li, M. Welling, and R. Zemel. 2015. "The variational fair autoencoder". *arXiv preprint arXiv:1511.00830*.

Louizos, C., K. Ullrich, and M. Welling. 2017. "Bayesian compression for deep learning". In: *Advances in Neural Information Processing Systems*. 3288–3298.

Louizos, C. and M. Welling. 2016. "Structured and efficient variational deep learning with matrix gaussian posteriors". In: *International Conference on Machine Learning*. 1708–1716.

Louizos, C. and M. Welling. 2017. "Multiplicative normalizing flows for variational Bayesian neural networks". In: *International Conference on Machine Learning*. 2218–2227.

Maaløe, L., C. K. Sønderby, S. K. Sønderby, and O. Winther. 2016. "Auxiliary deep generative models". In: *International Conference on Machine Learning*.

Maddison, C. J., A. Mnih, and Y. W. Teh. 2017. "The concrete distribution: A continuous relaxation of discrete random variables". *International Conference on Learning Representations*.

Makhzani, A., J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. 2015. "Adversarial autoencoders". *arXiv preprint arXiv:1511.05644*.

Mansimov, E., E. Parisotto, J. L. Ba, and R. Salakhutdinov. 2015. "Generating images from captions with attention". *arXiv preprint arXiv:1511.02793*.

Miao, Y., L. Yu, and P. Blunsom. 2016. "Neural variational inference for text processing". In: *International Conference on Machine Learning*. 1727–1736.

Mnih, A. and K. Gregor. 2014. "Neural variational inference and learning in belief networks". In: *International Conference on Machine Learning*.

Mnih, A. and D. Rezende. 2016. "Variational Inference for Monte Carlo Objectives". In: *International Conference on Machine Learning*. 2188–2196.

Mohamed, S. and D. J. Rezende. 2015. "Variational information maximisation for intrinsically motivated reinforcement learning". In: *Advances in Neural Information Processing Systems*. 2125–2133.

Molchanov, D., A. Ashukha, and D. Vetrov. 2017. "Variational dropout sparsifies deep neural networks". In: *International Conference on Machine Learning*. 2498–2507.

Naesseth, C., F. Ruiz, S. Linderman, and D. Blei. 2017. "Reparameterization gradients through acceptance-rejection sampling algorithms". In: *Artificial Intelligence and Statistics*. 489–498.

Neal, R. 2011. "MCMC Using Hamiltonian Dynamics". *Handbook of Markov Chain Monte Carlo*: 113–162.

Neal, R. M. and G. E. Hinton. 1998. "A view of the EM algorithm that justifies incremental, sparse, and other variants". In: *Learning in Graphical Models*. Springer. 355–368.

Paisley, J., D. Blei, and M. Jordan. 2012. "Variational Bayesian Inference with stochastic search". In: *International Conference on Machine Learning*. 1367–1374.

Papamakarios, G., I. Murray, and T. Pavlakou. 2017. "Masked autoregressive flow for density estimation". In: *Advances in Neural Information Processing Systems*. 2335–2344.

Pritzel, A., B. Uria, S. Srinivasan, A. P. Badia, O. Vinyals, D. Hassabis, D. Wierstra, and C. Blundell. 2017. "Neural episodic control". In: *International Conference on Machine Learning*. 2827–2836.

Pu, Y., Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin. 2016. "Variational autoencoder for deep learning of images, labels and captions". In: *Advances in Neural Information Processing Systems*. 2352–2360.

Ranganath, R., S. Gerrish, and D. Blei. 2014. "Black Box Variational Inference". In: *International Conference on Artificial Intelligence and Statistics*. 814–822.

Ranganath, R., D. Tran, and D. Blei. 2016. "Hierarchical variational models". In: *International Conference on Machine Learning*. 324–333.

Ravanbakhsh, S., F. Lanusse, R. Mandelbaum, J. Schneider, and B. Poczos. 2017. "Enabling dark energy science with deep generative models of galaxy images". In: *AAAI Conference on Artificial Intelligence*.

Rezende, D. J., S. Mohamed, I. Danihelka, K. Gregor, and D. Wierstra. 2016a. "One-shot generalization in deep generative models". In: *International Conference on International Conference on Machine Learning*. 1521–1529.

Rezende, D. J., S. Mohamed, and D. Wierstra. 2014. "Stochastic backpropagation and approximate inference in deep generative models". In: *International Conference on Machine Learning*. 1278–1286.

Rezende, D. J., S. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. 2016b. "Unsupervised learning of 3d structure from images". In: *Advances In Neural Information Processing Systems*. 4997–5005.

Rezende, D. and S. Mohamed. 2015. "Variational inference with normalizing flows". In: *International Conference on Machine Learning*. 1530–1538.

Roeder, G., Y. Wu, and D. K. Duvenaud. 2017. "Sticking the landing: Simple, lower-variance gradient estimators for variational inference". In: *Advances in Neural Information Processing Systems*. 6925–6934.

Rosca, M., B. Lakshminarayanan, and S. Mohamed. 2018. "Distribution matching in variational inference". *arXiv preprint arXiv:1802.06847*.

Roweis, S. 1998. "EM algorithms for PCA and SPCA". *Advances in Neural Information Processing Systems*: 626–632.

Ruiz, F. R., M. T. R. AUEB, and D. Blei. 2016. "The generalized reparameterization gradient". In: *Advances in Neural Information Processing Systems*. 460–468.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1988. "Learning representations by back-propagating errors". *Cognitive Modeling*. 5(3): 1.

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.* 2015. "Imagenet large scale visual recognition challenge". *International Journal of Computer Vision*. 115(3): 211–252.

Salakhutdinov, R. and H. Larochelle. 2010. "Efficient learning of deep Boltzmann machines". In: *International Conference on Artificial Intelligence and Statistics*. 693–700.

Salimans, T. 2016. "A structured variational auto-encoder for learning deep hierarchies of sparse features". *arXiv preprint arXiv:1602.08734*.

Salimans, T., D. P. Kingma, and M. Welling. 2015. "Markov Chain Monte Carlo and Variational Inference: Bridging the Gap." In: *International Conference on Machine Learning*. Vol. 37. 1218–1226.

Salimans, T. and D. A. Knowles. 2013. "Fixed-Form variational posterior approximation through stochastic linear regression". *Bayesian Analysis*. 8(4).

Semeniuta, S., A. Severyn, and E. Barth. 2017. "A hybrid convolutional variational autoencoder for text generation". *arXiv preprint arXiv:1702.02390*.

Serban, I. V., A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. Courville, and Y. Bengio. 2017. "A hierarchical latent variable encoder-decoder model for generating dialogues". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI Press. 3295–3301.

Sohl-Dickstein, J., E. Weiss, N. Maheswaranathan, and S. Ganguli. 2015. "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International Conference on Machine Learning*. 2256–2265.

Sønderby, C. K., T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. 2016a. "How to train deep variational autoencoders and probabilistic ladder networks". In: *International Conference on Machine Learning*.

Sønderby, C. K., T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther. 2016b. "Ladder variational autoencoders". In: *Advances in Neural Information Processing Systems*. 3738–3746.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. "Going deeper with convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1–9.

Tomczak, J. M. and M. Welling. 2016. "Improving variational auto-encoders using householder flow". *arXiv preprint arXiv:1611.09630*.

Tomczak, J. M. and M. Welling. "Improving variational auto-encoders using convex combination linear inverse autoregressive flow". In: *Benelearn 2017: Proceedings of the Twenty-Sixth Benelux Conference on Machine Learning, Technische Universiteit Eindhoven, 9-10 June 2017*. 162.

Tran, D., M. D. Hoffman, R. A. Saurous, E. Brevdo, K. Murphy, and D. M. Blei. 2017. "Deep probabilistic programming". *International Conference on Learning Representations*.

Tran, D., R. Ranganath, and D. M. Blei. 2015. "The variational Gaussian process". *arXiv preprint arXiv:1511.06499*.

Van den Oord, A., N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, *et al.* 2016. "Conditional image generation with PixelCNN decoders". In: *Advances in neural information processing systems*. 4790–4798.

Van Oord, A., N. Kalchbrenner, and K. Kavukcuoglu. 2016. "Pixel Recurrent Neural Networks". In: *International Conference on Machine Learning*. 1747–1756.

Vincent, P., H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. 2010. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion". *Journal of Machine Learning Research*. 11(Dec): 3371–3408.

Watter, M., J. Springenberg, J. Boedecker, and M. Riedmiller. 2015. "Embed to control: A locally linear latent dynamics model for control from raw images". In: *Advances in Neural Information Processing Systems*. 2746–2754.

White, T. 2016. "Sampling Generative Networks: Notes on a Few Effective Techniques". *arXiv preprint arXiv:1609.04468*.

Williams, R. J. 1992. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". *Machine Learning*. 8(3-4): 229–256.

Wingate, D. and T. Weber. 2013. "Automated variational inference in probabilistic programming". *arXiv preprint arXiv:1301.1299*.

Xu, W., H. Sun, C. Deng, and Y. Tan. 2017. "Variational autoencoder for semi-supervised text classification". In: *AAAI*. 3358–3364.

Yan, X., J. Yang, K. Sohn, and H. Lee. 2016. "Attribute2image: Conditional image generation from visual attributes". In: *European Conference on Computer Vision*. Springer. 776–791.

Yang, Z., Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick. 2017. "Improved variational autoencoders for text modeling using dilated convolutions". In: *International Conference on Machine Learning*. 3881–3890.

Zhao, T., R. Zhao, and M. Eskenazi. 2017. "Learning discourse-level diversity for neural dialog models using conditional variational autoencoders". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 654–664.