# Advances in Variational Inference

Vaibhav Vardhan[1]

***Abstract*— Focuses upon approximated inference in probabilistic machine learning, Variational Inference. Covers the advancements from the initial Mean Field VI to the currently-under exploration of Black Box VI and Amortized VI. Finally an implementation of Variational Auto-encoder over the MNIST dataset.**

## I. INTRODUCTION

The Variational Inference methodis one of the main methods used to tackle the Bayesian inference problem, and as an alternative to MCMC and Gibbs sampling, and in some sense are an extension of the EM algorithm. On one end, MCMC is purely a sampling based approach, and on the other, Variational Inference is an aproximation approach. VI involves finding the best approximation among a parametrised family through an optimization process.

### A. Mean Field Approach

In the mean field approach, the components of a random vector are all assumed to be independent i.e. the posterior approximation factorizes over the parameters. Hence, for a n-dimensional random variable z, the distribution can be represented as:

$$q(z_1, z_2, ..., z_n) = \prod_{i=1}^{n} q_i(z_i)$$

Here, each of the densities $f_i$ are themselves distributions with mean and variance parameters. The optimization in this case needs to be done over all the set of parameters for each of the $f_i$

### B. Drawbacks: Mean Field Approach

As obvious from the assumption, this method is only an approximation to the real quantity, an approximation made to make the posterior solvable. While making this assumption, since only the normal parameters are in consideration, we lost the dependent co-variance, in mathematical terms, the off-diagonal elements of the co-variance matrix. Hence, in general, an approach to go beyond this mean field approximation would yield closer to actual results, but for computational simplicity this method is necessary. Also, the coordinate descent optimization algorithm is too simplistic.

## II. BLACK BOX VARIATIONAL INFERENCE

### A. Background: KL Divergence

Kullback-Leibler Divergence :The parameter measures the closeness of two distributions.

$$KL(q||p) = \int_z q(z).log\frac{q(z)}{p(z|x)} = \mathbb{E}[log\frac{q(z)}{p(z|x)}]$$

### B. Background: Evidence Lower Bound

As mentioned earlier, when performing VI, we are minimizing the KL divergence between some distribution of random variable and another one that is easier to work with.

We define the Evidence Lower Bound as a function that would be minimized instead of KL (since both are equal upto a constant) for simplicity in evaluation.

Deriving the ELBO.

Jensens inequality:

$$f(\mathbb{E}[X]) \geq \mathbb{E}[f(X)]$$

Jensens inequality is applied to the log probability of the observations to get the ELBO.

$$log(\mathbb{E}_q\frac{p(x,z)}{q(z)}) \geq \mathbb{E}_q[log(x,z)] - \mathbb{E}[logq(z)]$$

The RHS denotes the ELBO for a probability model p(x,z) and approximation q(z) to the posterior. The quantity is always less than or equal to the evidence.

### C. General Idea

For the Bayesian inference, VI approximation is one of the most powerful tools available. But for any approximation, the method requires extensive mathematics to come up with the variational posterior, to derive the ELBO and gradients for updation. This process needs to done for every model implementation. BBVI offers a solution to tackle this issue. The 'black-box' name is derived from the fact that this generalized

Part of summer project 'probabilistic machine learning'

[1] vvgupta at iitk.ac.in

algorithm doesn't require the user to evaluate and change its inside working.

BBVI uses sampling based approximation of the gradient utilizing Monte Carlo simulation. And the bound is optimized using stochastic optimization.

### D. Derivation

BBVI maximizes the ELBO over $\phi$ and the only assumption is that $q_\phi$ is differentiable in its parameters $\phi$.

The lower bound for BBVI is:

$$\mathbb{L}(\lambda) = \mathbb{E}_{z-q\lambda}[(logp(x,z) - logq(z|\lambda))]$$

$\lambda$ represents the free parameters. Gradients are evaluated as:

$$\nabla\mathbb{L}(\lambda) = \mathbb{E}_{z-q\lambda}[\nabla_\lambda logq(z|\lambda)(logp(x,z) - logq(z|\lambda))]$$

The monte carlo approximation is:

$$\nabla\mathbb{L}(\lambda) \approx \frac{1}{S}\Sigma_{s=1}^{S}[\nabla_\lambda logq(z_s|\lambda)(logp(x,z_s) - logq(z_s|\lambda))]$$

where $z \sim q_\lambda(z)$

This ELBO can be optimized using a gradient based method. But the variance of these gradients estimates is generally to high to work and therefore this method relies on variance reduction techniques to reduce this variance.

### E. Variance Reduction

a. Centring and Variance Normalization: Writing the gradient in form of expectation allows a quantity to be subtracted from it, provided independent of z. Hence, due to this independence, a systematically calculated function of the data when subtracted can help to brind down the noise. Variance of the learning signal is also used to normalize the gradients. b. Rao-Blackwellization

## III. Variational Auto-encoders

In a nutshell, variational auto-encoders are generative models that estimate the probability density function of the training data. A variational auto-encoder, in general, consists of two parts, the encoder, the decoder and loss function. If visualized in the neural network setting, the encoder part of the network takes the input data, and through a deep network (configuration varies), maps it to a latent space. Latent space is essentially the entire data being encoded into a very small number of variables, also called the bottleneck of the entire network.

This lower dimensional space is actually stochastic, i.e. a gaussian probability density in the variational space. A random sampling from this space results in a noisy represntation through the decoder. On to the decoder itslef, in the neural network setting, it is just the opposite of the encoder (as the name suggests), and it maps the latent space that encoder outputs to a full representation input data.

Importance of such a set of network is highlighted in its use cases, with the ability to map a larger space successfully in lower-dimensional space, and vice-versa, it opens up the extensive world of extreme data compression, dealing with sparse-data, and the most exciting of all artificially generated data from the latent space sampling which behaves and looks like the real data since extracts its knowledge from the real-data.

### A. SGVB estimator

The evidence lower bound is a bound on the log probability of the data. But there is no straightforward way to compute the ELBO, since it requires taking an expectation over the variational posterior. Therefore we need a procedure to estimate the ELBO (more specifically we need some way to estimate the gradient of the ELBO so we can optimize it).

Hence the SGVB estimator is introduced:

$$logp(x) \geq \mathbb{E}_{q_\phi(z|x}[logp_\theta(x|z) - KL(q_\phi(z|x)||p(z)]$$

The first term, called as the reconstruction error, is the log likelihood of the observed x given the z sampled. It attempts to reconstruct x given the z in latent space.

The second term, as defined earlier is the KL divergence between q and the prior p. It works as a regularization term by forcing the z to look gaussian, and avoids an identity mapping.

### B. Reparametrization Trick

As the most general convention stands, back-propagation is required to train a neural network. And since here the encoder and decoder are both represented as networks, a training procedure is required, but due to the stochasticity of the node, training is not virtually not possible. Hence, a trick of reparametrization is introduced.

Details of this trick follows. Here the gradient of the following quantity is required:

$$\mathbb{E}_{p(z)}[f_\theta(z)]$$

$$\epsilon \sim p(\epsilon)$$

$$z = g_\theta(\epsilon, x)$$

$$\mathbb{E}_{p_\theta(z)}[f(z^i)] = \mathbb{E}_{p(\epsilon)}[f(g_\theta(\epsilon, x^i))]$$

$$\nabla_\theta \mathbb{E}_{p_\theta(z)}[f(z^i)] = \mathbb{E}_{p(\epsilon)}[f(\nabla_\theta g_\theta(\epsilon, x^i))]$$

$$\approx \frac{1}{L} \Sigma_{l=1}^{L} f(\nabla_\theta g_\theta(\epsilon^l, x^i))$$

We use the reparameterization trick to express a gradient of an expectation (1) as an expectation of a gradient (2). Provided g is differentiable—something Kingma emphasizes—then we can then use Monte Carlo methods to estimate the required quantity. This approach has much lower variance than the score function estimator, and will enable us to learn models that we otherwise couldn't learn.

### *C. Auto-encoding variational Bayes (AEVB)*

The AEVB algorithm is simply the combination of (1) the auto-encoding ELBO reformulation, (2) the black-box variational inference approach, and (3) the reparametrization-based low-variance gradient estimator. It optimizes the auto-encoding ELBO using black-box variational inference with the reparametrized gradient estimator. This algorithm is applicable to any deep generative model with latent variables that is differentiable in $\theta$.

The variational autoencoder can be intepreted as a directed latent-variable probabilistic graphical model. It can also be viewed as a particular objective for training an auto-encoder neural network; unlike previous approaches, this objective derives reconstruction and regularization terms from a more principled, Bayesian perspective.

### *D. Comparison*

The advantages and disadvantages would essentially be extracted from the content above itself, hence, redundant to include.

### *E. Experiment*

The AEVB was implemented on the MNSIT dataset and the learnt latent variables were visualised.

Parameters of the experiment were:

- No. of latent variables: 2
- Network structure (Encoder): Convolution layers followed by a Dense layer
- Network structure (Decoder): Encoder reversed
- Image Size: 24*24
- Optimizer: RMSProp

Visualizations of the latent space and digits is included.

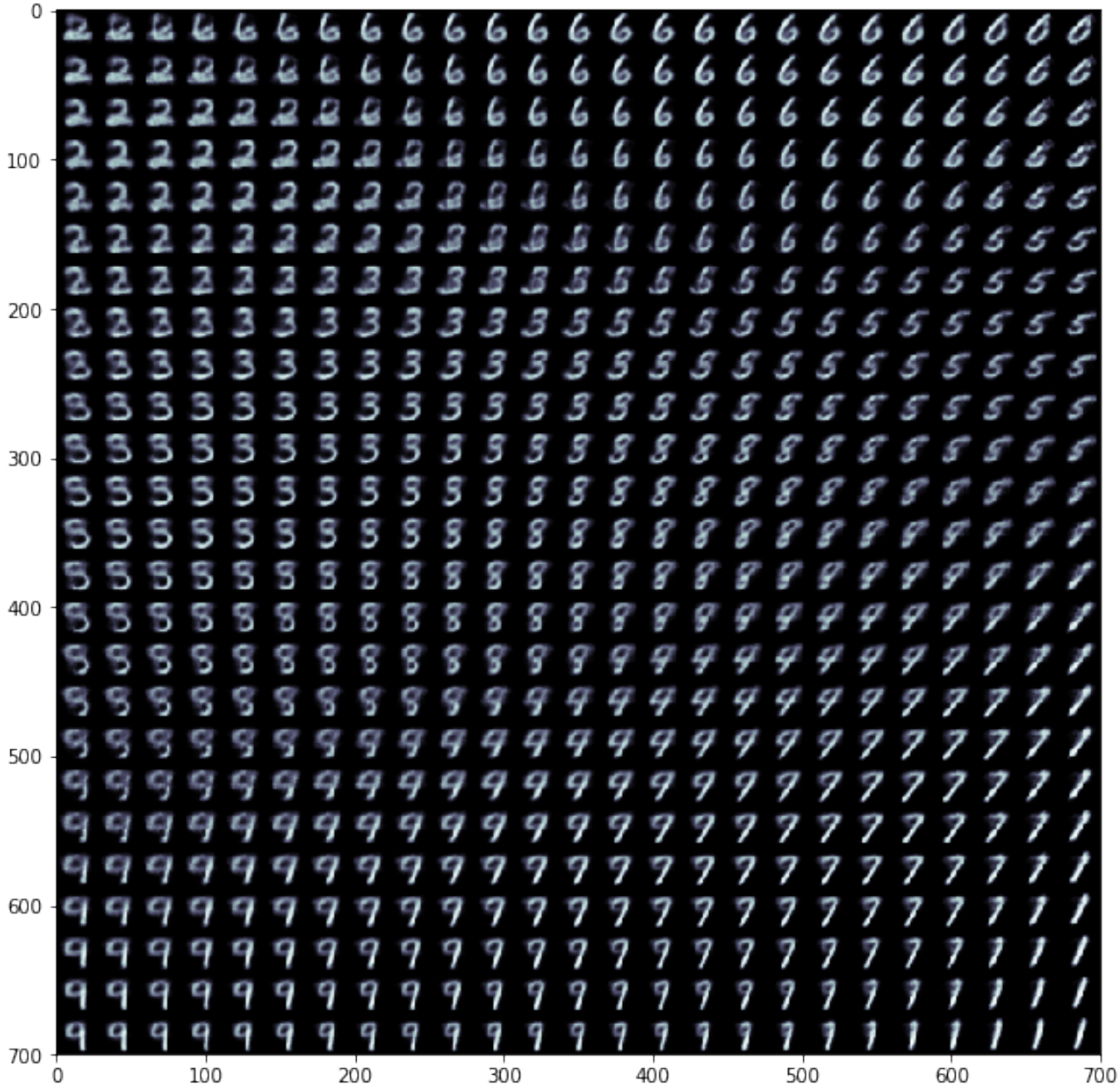


Fig. 1. MNIST Uniformly Sampled

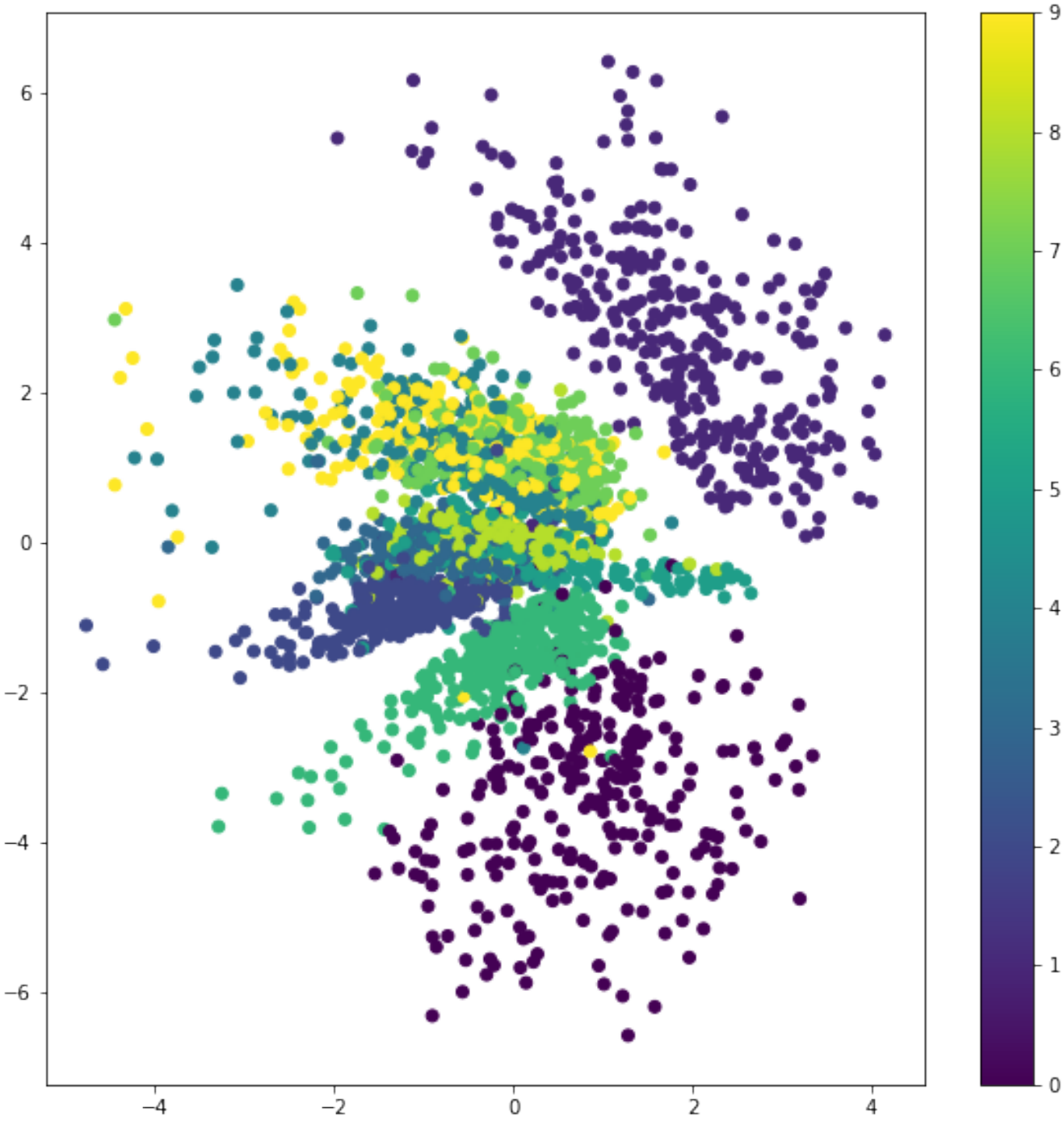


Fig. 2. Latent Variable Visualization