University of Cambridge

MLSALT 4: Auto-Encoding Variational Bayes

Thomas F. W. Nicholson, tfwn2, Pembroke College
Will Tebbutt, wct23, Darwin College
Paweł Budzianowski, pfb30, Clare Hall College

1. Introduction

This paper[5] concerns itself with the scenario in which we wish to find a point-estimate to the parameters θ of some parametric model in which we generate each observations $\mathbf{x}_i$ by first sampling a "local" latent variable $\mathbf{z}_i \sim p_\theta(\mathbf{z})$ and then sampling the associated observation $\mathbf{x}_i \sim p_\theta(\mathbf{x} \mid \mathbf{z})$. The conditional independence assumptions in this model are shown in the graphical model in 1.

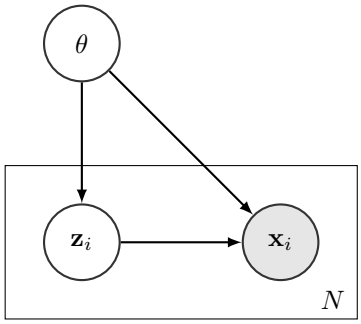

Figure 1: Directed graphical model representing the conditional independencies in the proposed problem scenario. Each i^{th} observation is conditionally independent given the model parameters θ.

The posterior $p_\theta(\mathbf{z}_i \mid \mathbf{x}_i)$ is intractable for a continuous latent space whenever either the prior $p_\theta(\mathbf{z}_i)$ or the likelihood $p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)$ are non-Gaussian, meaning that approximate inference is required. To this end Autoencoding Variational Bayes makes two contributions in terms of methodology, introducing a differentiable stochastic estimator for the variational lower bound to the model evidence, and using this to learn a recognition model to provide a fast method to compute an approximate posterior distribution over "local" latent variables given observations.

In this report we will first discuss the methodological contributions of the Autoencoding Variational Bayes paper, with care taken to treat each of the afformentioned methodological contributions separately. We then discuss their use of this framework to derive the Variational Autoencoder, a model exactly of the form described above with a Gaussian prior, multi-layer perceptron (MLP) parameterised likelihood and MLP-parameterised recognition model. We reproduce the key experiments from the original paper and conduct several more including investigating the sensitivity of the reported results to the network architecture and whether the Variational Autoencoder provides improved reconstructive performance over a traditional "vanilla" autoencoder with the same architecture.

2. Stochastic Variational Inference

The aim of variational inference is to provide a deterministic approximation to an intractable posterior distribution by finding parameters ϕ such that $\mathrm{D}_{KL}(q_\phi(\theta) \mid\mid p(\theta \mid D))$ is minimised. This is achieved by noting that

$$\begin{aligned}\mathrm{D}_{KL}(q_\phi(\theta) \mid\mid p(\theta \mid D)) &= \log p(D) + \mathbb{E}_{q_\phi(\theta)}[\log q_\phi(\theta) - \log p(\theta, D)] \\ &=: \log p(X) - \mathcal{L}(\phi; D). \end{aligned} \tag{1}$$

Noting that $\log p(D)$ is constant w.r.t. ϕ, we can now minimise the KL-divergence by maximising the evidence lower bound (ELBO) $\mathcal{L}$ (that this is indeed a lower bound follows from the non-negativity of the KL-divergence).

Aside from some notable exceptions (eg. [7]) this quantity is not tractably point-wise evaluable. However, if $q_\phi(\theta)$ and $\log p(\theta, D)$ are point-wise evaluable, it can be approximated using Monte Carlo as

$$\mathcal{L}(\phi; D) \approx \frac{1}{L} \sum_{l=1}^{L} \log p(\theta_l, D) - \log q_\phi(\theta_l), \quad \theta_l \sim q_\phi(\theta) \tag{2}$$

This stochastic approximation to the ELBO is not differentiable w.r.t. ϕ as the distribution from which each θ_l is sampled itself depends upon ϕ, meaning that the gradient of the log likelihood cannot be exploited to perform inference. One of the primary contributions of the paper being reviewed is to provide a differentiable estimator for $\mathcal{L}$ that allows gradient information in the log likelihood $p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)$ to be exploited, resulting in an estimator with lower variance. In particular it notes that if there exists a tractable reparameterisation of the random variable $\tilde{\theta} \sim q_\phi(\theta)$ such that

$$\tilde{\theta} = g_\phi(\epsilon), \quad \epsilon \sim p(\epsilon), \tag{3}$$

then we can approximate the gradient of the ELBO as

$$\mathcal{L}(\phi; D) = \mathbb{E}_{p(\epsilon)}[\log p(\theta, X) - q_\phi(\theta)] \approx \frac{1}{L} \sum_{l=1}^{L} \log p(\theta_l, X) - \log q_\phi(\theta_l) =: \tilde{\mathcal{L}}^1(\phi; X), \tag{4}$$

where $\theta_l = g_\phi(\epsilon_l)$ and $\epsilon_l \sim p(\epsilon)$. Thus the dependence of the sampled parameters θ on ϕ has been removed, yielding a differentiable estimator provided that both q_ϕ and $\log p(\theta, D)$ are themselves differentiable. Approximate inference can now be performed by computing the gradient of $\tilde{\mathcal{L}}^1$ w.r.t. ϕ either by hand or using one's favourite reverse-mode automatic differentiation package (eg. Autograd [6]) and performing gradient-based stochastic optimisation to maximise the elbo using, for example, AdaGrad [2].

The authors also point out that one can re-express the elbo in the following manner

$$\mathcal{L}(\phi; D) = \mathbb{E}_{q_\phi(\theta)}[\log p(D \mid \theta)] - \mathrm{D}_{KL}(q_\phi(\theta) \mid\mid p(\theta)). \tag{5}$$

This is useful as the KL-divergence between the variational approximation $q_\phi(\theta)$ and the prior over the parameters θ has a tractable closed-form expression in a number of useful cases. This leads to a second estimator for the elbo:

$$\tilde{\mathcal{L}}^2(\phi; D) := \frac{1}{L} \sum_{l=1}^{L} \log p(D \mid \theta_l) - \mathrm{D}_{KL}(q_\phi(\theta) \mid\mid p(\theta)). \tag{6}$$

It seems probable that this estimator will in general have lower variance than $\tilde{\mathcal{L}}^1$.

So far Stochastic Variational Inference has been discussed only in a general parametric setting. The paper's other primary contribution is to use a differentiable recognition network to learn to parameterise the posterior distribution over latent variables z_i local to each observation x_i in a parametric model. In particular, they assume that given some global parameters θ, $\mathbf{z}_i \sim p_\theta(\mathbf{z})$ and $\mathbf{x}_i \sim p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)$. In the general case the posterior distribution over each z_i will be intractable. Furthermore, the number of latent variables $\mathbf{z}_i$ increases as the number of observations increases, meaning that under the framework discussed above we would have to optimise the variational objective with respect to each of them independently. This is potentially computationally intensive and quite wasteful as it completely disregards any information about the posterior distribution over the z_i provided by the similarities between inputs locations $\mathbf{x}_{\neq i}$and corresponding posteriors $\mathbf{z}_{\neq i}$. To rectify this the recognition model $q_\phi(\mathbf{z} \mid \mathbf{x})$ is introduced.

Given the recognition model and a point estimate for θ, the ELBO becomes

$$\begin{aligned}\mathcal{L}(\theta, \phi; D) &= \mathbb{E}_{q_\phi(\mathbf{z_1}), \ldots, q_\phi(\mathbf{z}_N)}\left[\log \prod_{i=1}^{N} p_\theta(\mathbf{x}_i, \mathbf{z}_i) - \log \prod_{i=1}^{N} q_\phi(\mathbf{z})_i\right] \\ &= \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{z}_i)}[\log p_\theta(\mathbf{x}_i, \mathbf{z}_i) - \log q_\phi(\mathbf{z}_i)]\end{aligned} \tag{7}$$

For this ELBO we can derive a similar result to $\tilde{\mathcal{L}}^1$, where we do not assume a closed-form solution for the KL divergence between distributions and include mini-batching to obtain an estimator for the ELBO for a mini-batch of observations

$$\tilde{\mathcal{L}}^A(\theta, \phi; D) \approx \frac{N}{LM} \sum_{i=1}^{M} \sum_{l=1}^{L} \log p_\theta(\mathbf{x}_i, \mathbf{z}_{i,l}) - \log q_\phi(\mathbf{z}_{i,l}), \quad \mathbf{z}_{i,l} = g_\phi(\mathbf{x}_i, \epsilon_{i,l}), \ \epsilon_{i,l} \sim p(\epsilon) \tag{8}$$

where the M observations in the mini-batch are drawn uniformly from the data set comprised of N observations and for each observation we draw L samples from the approximate posterior $q_\phi(\mathbf{z}_i \mid \mathbf{x}_i)$. Similarly, if $q_\phi(\mathbf{z} \mid \mathbf{x})$ and $p_\theta(\mathbf{z})$ are such that the KL-divergence between them has a tractable closed-form solution then we can use an approximate bound which we could reasonably expect to have a lower variance:

$$\begin{aligned}\mathcal{L}(\theta, \phi; D) &= \mathbb{E}_{q_\phi(\mathbf{z}_1),\ldots,q_\phi(\mathbf{z}_N)}\left[\log \prod_{i=1}^{N} p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)\right] - \mathrm{D}_{KL}\left(\prod_{i=1}^{N} q_\phi(\mathbf{z}_i \mid \mathbf{x}_i) \;||\; \prod_{i=1}^{N} p_\theta(\mathbf{z}_i)\right) \\ &= \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{z}_i \mid \mathbf{x}_i)}\left[\log p_\theta(\mathbf{x}_i \mid \mathbf{z}_i)\right] - \mathrm{D}_{KL}\left(q_\phi(\mathbf{z}_i \mid \mathbf{x}_i) \;||\; p_\theta(\mathbf{z}_i)\right) \\ &\approx \frac{N}{M} \sum_{i=1}^{M}\left[\frac{1}{L}\sum_{l=1}^{L} \log p_\theta(\mathbf{x}_i \mid \mathbf{z}_{i,l}) - \mathrm{D}_{KL}\left(q_\phi(\mathbf{z}_i \mid \mathbf{x}_i) \;||\; p_\theta(\mathbf{z}_i)\right)\right] =: \tilde{\mathcal{L}}^B(\theta, \phi; D), \end{aligned} \tag{9}$$

where $z_{i,l} = g_\phi(\mathbf{x}_i, \epsilon_{i,l})$ and $\epsilon_{i,l} \sim p(\epsilon)$.

3. The Variational Autoencoder

The variational autoencoder exploits the methods described in the previous section to define a probabilistic model whose elbo is highly reminiscent of the objective optimised in a traditional autoencoder. In particular we define a generative model where we assume that the i^{th} observation was generated by first sampling a latent variable $\mathbf{z}_i \sim \mathcal{N}(0, I)$ and that the each observation vector real-valued observation $\mathbf{x}_i \sim \mathcal{N}\left(\mu_\theta(\mathbf{z}_i), \sigma^2_\theta(\mathbf{z}_i)\right)$ where

$$\mu_\theta(\mathbf{z}_i) = \mathbf{h}_i W^{(dec)}_\mu + \mathbf{b}^{(dec)}_\mu, \tag{10}$$

$$\log \sigma^2_\theta(\mathbf{z}_i) = \mathbf{h}_i W^{(dec)}_\sigma + \mathbf{b}^{(dec)}_\sigma, \tag{11}$$

$\mathbf{h}_i \in \mathbb{R}^{1 \times D_h}$ is the output at the final hidden layer of the "decoder MLP", $W^{(dec)}_\mu, W^{(dec)}_\sigma \in \mathbb{R}^{D_h \times D_z}$ are matrices mapping from the D_h hidden units to the D_z dimensional latent space. Similarly, $\mathbf{b}^{(dec)}_\mu, \mathbf{b}^{(dec)}_\sigma \in \mathbb{R}^{1 \times D_z}$ are row vector biases. Note that the variances are parameterised implicitely through their logs to ensure that they are correctly valued only on the positive reals.

If the output vectors are binary valued then $x_i \sim \text{Bernoulli}(f_\theta(\mathbf{z}_i))$ where again given $\mathbf{h}_i$, the output at the final hidden layer of the decoder MLP for latent-space value $\mathbf{z}_i$

$$f_\theta(\mathbf{z}_i) = \left[1 + \exp\left(-\mathbf{h}_i W^{(dec)} - \mathbf{b}^{(dec)}\right)\right]^{-1}, \tag{12}$$

where $W^{(dec)} \in \mathbb{R}^{D_h \times D_z}$ and $b^{(dec)} \in \mathbb{R}^{1 \times D_z}$.

The recognition model is given by

$$q_\phi(\mathbf{z} \mid \mathbf{x}) = \mathcal{N}(\mathbf{z}; \mu_\phi, \mathbf{x}_i, \sigma_{\phi^2}(\mathbf{x}_i)), \tag{13}$$

where the distributional parameters $\mu_\phi(\mathbf{x}_i)$ and $\sigma^2_\phi(\mathbf{x}_i)$ are again given by an MLP, which will be refered to as the "encoding MLP", whose input is $\mathbf{x}_i$.

In this case, there is a tractable closed form solution for the KL-divergence between the variational posterior $q_\phi(\mathbf{z} \mid \mathbf{x})$ and the prior $p_\theta(\mathbf{z})$ meaning that we can approximate the ELBO using equation 9. Additionally we have that

$$g_\phi(\mathbf{x}, \epsilon) = \mu_\phi(\mathbf{x}_i) + \sigma_\phi(\mathbf{x}_i) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \tag{14}$$

The algorithm to perform inference in the Variational Autoencoder is provided in algorithm 1.

4. Replications and Extensions

4.1. Visualisation of learned manifolds

It is possible to observe what the encoder learnt during training if we choose a low-dimensional latent space e.g. $2D$. Following the authors, we mapped the linearly spaced grid of coordinates over the unit square through the

Algorithm 1 Procedure by which inference is performed in the Variational Autoencoder.

$\theta, \phi \leftarrow$ Initialisation.
while not converged in θ, ϕ **do**
  Pick subset of size $\mathbf{x}_{1:M}$ from the full dataset uniformly at random.
  Compute $\mu_{1:M}, \log \sigma^2_{1:M}$ using the encoding MLP.
  For all $i \in \{1, ..., M\}$, sample $\epsilon_i \sim \mathcal{N}(0, I)$.
  For all $i \in \{1, ..., M\}$, $\mathbf{z}_i \leftarrow \mu_i + \sigma_i \odot \epsilon_i$.
  For all $i \in \{1, ..., M\}$, compute $\log p(\mathbf{x}_i \mid \mathbf{z}_i)$ using the decoder MLP and likelihood function.
  Compute $g \leftarrow \nabla_{\theta,\phi} \tilde{\mathcal{L}}^B(\theta, \phi; \mathbf{x}_{1:M}, \epsilon_{1:M})$
  Update θ, ϕ using noisy gradient estimate g.
end while

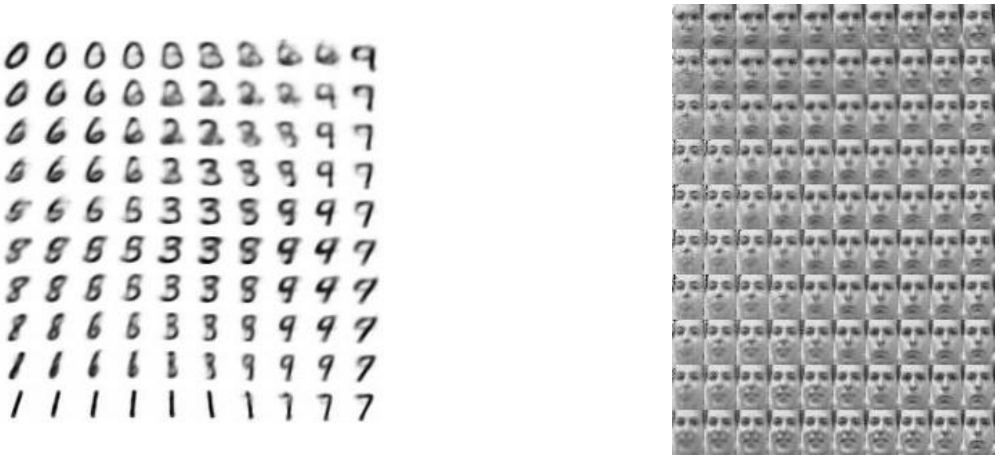

Figure 2: Visualisations of learned data manifolds using two-dimensional latent space, trained with AEVB and Gaussian prior over the latent variable $\mathbf{z}$.

inverse CDF of the Gaussian (our prior for the latent space is Gaussian) to obtain the value of $\mathbf{z}$. Then, we plotted the output of our decoder $p_\theta(\mathbf{x}|\mathbf{z})$ with the estimated parameters θ. Figure 2 shows the results for two learned manifolds.

In the case of the MNIST dataset, similarly to the results in the paper the central number is 3 which corresponds to the point $(0, 0)$. In the case of the second dataset, it is interesting that the first half of the manifold shows the face from left profile while the second half slowly transforms it to the right profile. Moreover, in both parts of the manifold we can observe happy and grim faces which is a different representation comparing to the one obtained by the authors, who found that happy faces and grim faces were split vertically.

4.2. Number of samples and a batch size

The authors found out that the number of samples L per data point can be set to 1 as long as the mini-batch size M was large enough, e.g. $M = 100$. However, they didn't provide any empirical results and we decided to run the comparison between the number of samples and a batch size in terms of optimizing the lower bound. Due to the computational requirements (32 models needed for evaluation), the experiment was run only with Frey Face dataset with 2000 epochs of training. Figure 3 presents the results for sample size ranging from 1 to 8 and batches of size 20, 60, 100 and 140.

The results are somewhat inconclusive and don't confirm the authors' findings. In the case of the test data, the highest score was obtained with $L = 2$ and $M = 100$ and the score was invariant to the sample size only with the batch size larger than 20. In the case of training set, the highest score for lower bound was obtained with the batch of size 20 where the sample size L makes a big influence. For larger batch sizes, the sample size becomes invariant in terms of the score for the lower bound. However, it is possible that the models trained with a larger batch size might need more time to converge hence further investigation is needed here.

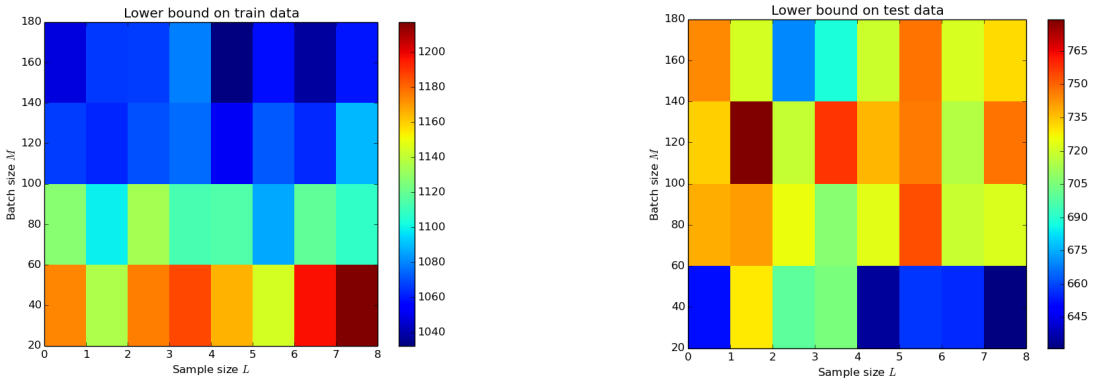

Figure 3: Heatmaps of lower bound for Frey Face data set.

4.3. Increasing the depth of the encoder

Extending the depth of the neural networks proved to be a very helpful method in increasing the power of neural architectures. The authors considered the encoder with only one hidden layer and we decided to add additional hidden layers to test how much gains can be obtained in terms of optimizing the lower bound and at the same time still obtaining robust results. The experiment was run with MNIST data set having 500 hidden units with the size of the latent space N_z set to 10 which seems to be optimal value.

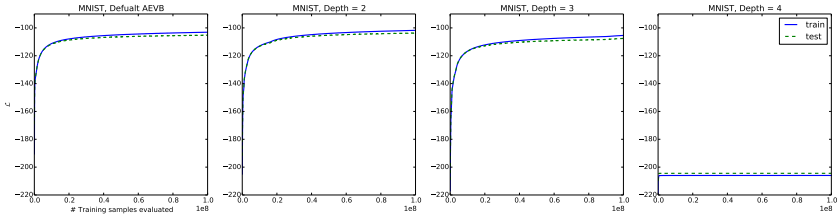

Figure 4: Comparison of performance between different encoders architectures in terms of optimizing the lower bound with dimensionality of latent a space set to 10.

Additional hidden layers didn't yield substantial increase in the performance however the encoder with two hidden layers performed slightly better than the original architecture. Presumably owing to the "vanishing gradients" problem, adding fourth hidden layer resulted in the inability of the network to learn.

4.4. Noisy KL-divergence estimate

Until now we were assuming that the KL-divergence term $D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z}))$ can be obtained using a closed-form expression. However, in the case of the non-Gaussian distributions it is often impossible and this term also requires estimation by sampling. This more generic SGVB estimator $\widetilde{\mathcal{L}}^A(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}^{(i)})$ is of the form:

$$\widetilde{\mathcal{L}}^A(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}_i) = \frac{1}{L} \sum_{l=1}^{L} \left(\log p_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{z}_{i,l}) - \log q_\phi(\mathbf{z}_{i,l}|\mathbf{x}_i) \right).$$

Naturally, this form in general will have greater variance than the previous estimator. We decided that it will be informative to compare the performance of both estimators using only one sample i.e. $L = 1$ – this will allow us to observe how much more robust is the first estimator. Figure 5 shows the results for the MNIST data set.

As we can see, in each case the performance of the $\widetilde{\mathcal{L}}^B$ estimator is substantially better. Moreover, this shows that the generic estimator might be used if we increase the sample size L to reduce the variance of the estimator, however, this needs to be examined empirically. Additionally, this comes with higher computational costs.

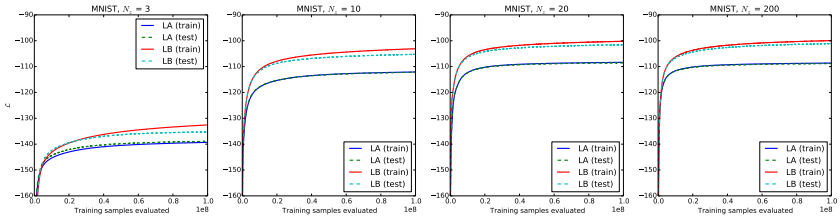

Figure 5: Comparison of two SGVB estimators in terms of optimizing the lower bound for different dimensionality of latent space (N_z).

4.5. Activation functions

The paper considered only tanh as an activation function in the case of hidden layers. That is why, we tested also sigmoid and ReLu activation functions for different size of latent variable space N_z to compare differences in terms of optimising lower bound and computational efficiency. Figure 6 shows results for three such set-ups with the size of the latent space N_z set to 10 for MNIST data set.

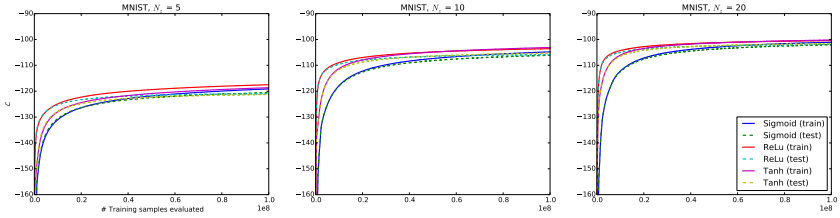

Figure 6: Comparison of different activation functions in terms of optimizing the lower bound.

Although, as it was expected, ReLu function learns the fastest, there is no substantial gains over tanh function. In each case, the sigmoid function learns the slowest and obtains the lowest score for the bound and at the same time the training time took about 20% more time than in the case of the two other functions.

5. Reconstruction

The VAE provides a probabilistic approach to modeling the data as a distribution around some underlying manifold. This allows us to define a posterior distribution over the latent space, and a distribution over input space which can be used to score inputs, while being able to measure uncertainty over both estimates. Aside from these traditional advantages of a Bayesian approach over traditional methods do we also gain advantages in terms of modeling capabilities?
The AE also takes an input vector and maps it to some latent space, however only providing a point estimate on some lower dimensional manifold. AEs are trained by minimizing the reconstruction error of training examples usually measured by mean squared error (MSE), and a weight regularizer:

$$E(\mathbf{x}, \theta) = \|\mathbf{x} + \sigma_\theta(\mathbf{x})\|^2 + \lambda \ell(\theta), \tag{15}$$

where $\sigma(\cdot)$ represents the application of the autoencoder (encoding followed by decoding) to a particular data example, $\ell(\cdot)$ represents a specific weight decay, and λ represents the weight penalty. This can be seen to resemble the variational lower bound:

$$\mathcal{L}\left(\theta, \phi; \mathbf{x}_i\right) = \mathbb{E}_{q_\phi(\mathbf{z}_i|\mathbf{x}_i)}\left[\log p_\theta\left(\mathbf{x}_i \,|\, \mathbf{z}_i\right)\right] - \mathrm{D}_{KL}\left(q_\phi\left(\mathbf{z}_i \,|\, \mathbf{x}_i\right) \;||\; p_\theta\left(\mathbf{z}_i\right)\right), \tag{16}$$

in which the first term can be seen as the expected negative reconstruction weight and the second acts as a regularizer pulling latent variables towards the prior. Rather than defining a distribution in latent variable space, the AE instead provides a point estimate in the bottleneck layer, but the two are analogous in that they provide a lower dimensional representation of an input.
Since the latent representation in the VAE is a distribution we have to choose how we use this to create a single decoded example. Both using the mean of the distribution and averaging the output of multiple samples were tried. Using the mean was shown to give an absolute improvement in MSE of about 0.2%, and so that is the method that is used for further experiments.

Instantiations of each model type were trained using the configurations: the encoder and decoder each have one hidden layer of size 500 consisting of, tanh activation functions and using the same ℓ_2 normalization penalty on the same portion of the MNIST data set. The resulting mean construction errors for the two models for a different size of the latent space/ compression layer are shown in Figure 7.

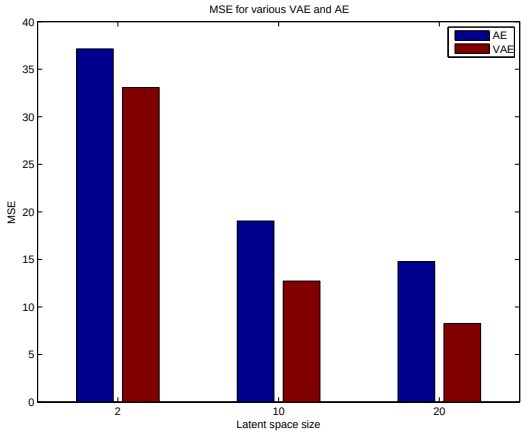

Figure 7: Comparison of reconstruction error measured by MSE for variational auto-encoder and vanilla auto-encoder for various sizes of representation space

It can be seen that the variational auto-encoder outperforms the regular auto-encoder for all reduced dimensional sizes. The difference in reconstruction error increases as the size of the latent space/ bottleneck layer increases. This suggests that the difference in performance is due to the better generalization afforded by the variational bayesian approach.
Note that the AE is directly trained to minimize the criterion that we have used to compare the two models, whereas VAE is trained to minimize the expected reconstruction rate, which for the discrete case is the binary cross entropy. So despited having an "unfair advantage" the auto-encoder still performs worse.
To further contrast the two approaches (Figure 8) shows specific examples of digits constructed by VAE and AE. It can be seen that generally the VAE produces images that are sharper and more similar to the original digit, with the exception of the number 9 for the 2 dimensional case. It was initially speculated that this would correspond to a higher variance of the posterior, however this was found to not be the case. Looking at the representation of the two dimensional latent space in ? we can see that the further right we go the more rightward slanting nines we have, so having a leftward slanting nine would push us away from nines towards the region of weight space containing eights. In such a compressed latent space it seems reasonable to assume that there will be forms of certain digits that are unrepresentable, in this case we have found an unfortunate example on which the VAE performs poorly.

Figure 8: Examples of the quality of reconstruction of certain digits for VAE and AE

These results must not be taken at face value because, although greatest care was taken to compare the two approaches on an even footing, it is possible that AE is more susceptible to hyperparameter settings, a full investigation of which was not performed here.

6. Full Variational Bayes

As well as providing a method of variational inference over the parameters of a latent space Kingma and Welling also detail a method of performing full variational Bayesian inference over the parameters. In this scheme we place a hyperprior over the parameters of the model $p_\alpha(\theta)$. The variational of the lower bound of the marginal likelihood can then be written:

$$\mathcal{L}\left(\phi; \mathbf{X}\right) = \mathbb{E}_{q_\phi(\theta)}\left[\log p_\theta(\mathbf{X})\right] - \mathrm{D}_{KL}\left(q_\phi(\theta) \;||\; p_\alpha\left(\theta\right)\right) \tag{17}$$

By maximizing this we are encouraging the model to reconstruct the data accurately, while constraining the form that the distribution of parameters can take. For a particular point we have a variational lower bound on

the marginal likelihood:

$$\mathcal{L}\left(\theta, \phi; \mathbf{x^{(i)}}\right) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x^{(i)}})}\left[\log p_\theta(\mathbf{z}|\mathbf{x^{(i)}})\right] - \mathrm{D}_{KL}\left(q_\phi(\mathbf{z}|\mathbf{x^{(i)}}) \;||\; p_\theta(\mathbf{z})\right) \tag{18}$$

Combining eq. (17) and eq. (18), using the same reparameterization trick as $\tilde{z} = g_\phi(\epsilon)$ with $\epsilon \sim p(\epsilon)$ and using the same trick for the variational approximation to the posterior over parameters: $\tilde{\theta} = h_\phi(\zeta)$ with $\zeta = p(\zeta)$ we arrive at the differentiable Monte Carlo estimate for the variational lower bound of the marginal likelihood:

$$\mathcal{L}(\phi; \mathbf{X}) \approx \frac{1}{L}\sum_{l=1}^{L} N \cdot (\log p_{\tilde{\theta}}(\mathbf{x}|\tilde{\mathbf{z}}) + \log p_{\tilde{\theta}}(\tilde{\mathbf{z}}) - \log q_\phi(\tilde{\mathbf{z}}|x)) + \log p_\alpha(\tilde{\theta}) - \log q_\phi(\tilde{\theta}) \tag{19}$$

which can be maximized by performing SGVB as before by differentiating with respect to ϕ.

The authors provides a concrete example of a realization of the above model in which we assume standard normal distributions for the priors over the variables and latent space, and have variational approximations to the posteriors of the form:

$$\begin{aligned} q_\phi(\theta) &= \mathcal{N}(\theta; \mu_\theta, \sigma_\theta^{\mathbf{2}}\mathbf{I}) \\ q_\phi(\mathbf{z}|\mathbf{x}) &= \mathcal{N}(\mathbf{z}; \mu_{\mathbf{z}}, \sigma_{\mathbf{z}}^{\mathbf{2}}\mathbf{I}) \end{aligned} \tag{20}$$

Those assumptions enable us to obtain closed form solutions for the KL term. This approach was implemented and tested on the MNIST and Frey Face data sets. Although the lower bound was increased, progress was extremely slow, the training lower bound increased much faster than the validation set, and evaluation of the reconstruction ability of the resulting models showed that no learning had taken place.

The very slow progress to an eventual poor model resembled the effects of starting in a poor region of neural network parameter space,and so the initial values of μ_σ were seeded with the MAP solutions from a regular VAE trained to convergence while σ_θ^2 were all set to be 10^{-3}, thereby hopefully encouraging the model to learn a distribution around a known good configuration of parameters. Nonetheless, this yielded identically poor results.

The purpose of performing inference over the parameters is to reduce overfitting and promote generalization. However in the scheme proposed it appears that the model underfits to the extent that it simply does not learn. There are a number of possible explanations for this. One problem that was faced in the implementation was negative values of variances. This was worked around by using a standard deviation which is then squared to yield a positive variance. In their recent paper on variational inference over MLP parameters [1] work around this by parameterizing σ as $\sigma = \log(1 + \exp(\rho))$. Despite yielding a closed form solution, a standard normal prior over weights is perhaps too wide for MLP weight parameters, which typically have very low variance about zero. [1] found that despite not yielding a closed form solution a complicated spike-and-slab-like prior performed best composed of a mixture of a high variance and low variance Gaussian centered at 0 performed well.

Performing full variational inference will allow robust weight estimates even in low resource environments. The approach in the paper favours a neat analytical form of prior over analytically complicated priors that may induce more reliable weight estimates. The trade off between precision of gradient estimates and efficacy of form is an interesting problem that requires further research.

7. Future Works

7.1. Latent space priors

An obvious extension to the paper to investigate is to simply change the form of the prior and the variational approximation in an attempt to induce a particular form of latent space. For example a particularly interesting set up would be to define a sparsity inducing prior that encourages each dimension of the latent space to be approximately valued on $\{0, 1\}$. An obvious choice would be a set of sparse Beta distributions (ie. ones in which the shape parameters $\alpha, \beta < 1$), but one could also use pairs of univariate Gaussians with means 0 and 1 and small variances.

Such a prior would be useful for two reasons - firstly it would allow one to provide a binary encoding for a data set by truncating the posterior approximation for any particular observation to be exactly vector binary valued allowing for a large amount of lossy compression. The posterior distribution over the parameters θ and

latent values $\mathbf{z}_i$ also contains rotational symmetry which may affect the quality of the approximate inference if it attempts to place posterior mass over the entirety of this. Were a prior such as the one proposed used, this rotational symmetry would be destroyed and replaced with a "permutation symmetry", similar to that found in a finite mixture model.

7.2. Non-parametric posterior approximation

We currently assume a simple parametric form for the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ that allows the use of the reparameterization trick. Although this yields a robust training regime, it limits the expressibility of the model to a subset of potential distributions. If instead we directly use the $g_\phi(\mathbf{x}, \epsilon)$ we can induce an arbitrarily complex posterior that would allow us to approximate any true posterior.
This idea has been recently realised using Gaussian processes by [8] who draw random latent input samples, push them through a non-linear mapping and then draw posterior samples. If we instead were to use a MLP to model $g_\phi(\mathbf{x}, \epsilon)$ we can, theoretically, model arbitrary posteriors. The problem now is the ability to yield a differentiable distribution over latent space which can potentially be sampling multiple $g_\phi(\mathbf{x}, \epsilon_i)$ to approximate a distribution, and batching gradients over all samples. This is akin to a Monte Carlo estimate of the variational posterior.

7.3. Scheduled VAEB

One of the most popular approaches in the unsupervised learning using autoencoding structures is making the learned representation robust to partial corruption of the input pattern [9]. This also proved to be an effective step in pre-training of deep neural architectures. Moreover, this method can be extended where the network is trained with a schedule of gradually decreasing noise levels [3]. This approach is motivated by a desire to encourage the network to learn a more diverse set of features from a coarse-grained to fine-grained ones.

Moreover, there was recently an effort to inject noise into both an input and in the stochastic hidden layer (denoising variational autoencoder, DVAE) which yields better score in terms of optimising the log likelihood [4]. In order to estimate the variational lower bound the corrupted input $\widetilde{\mathbf{x}}$, obtained from a known corruption distribution $p(\widetilde{\mathbf{x}}|\mathbf{x})$ around $\mathbf{x}$, requires to be integrated out which is intractable in the case of $\mathrm{E}_{p(\widetilde{\mathbf{x}}|\mathbf{x})}\left[q_\phi(\mathbf{z}|\widetilde{\mathbf{x}})\right]$. Thus, Im et al. arrived at the new form of the objective function – the denoising variational lower bound:

$$\mathcal{L}^C \stackrel{\text{def}}{=} \mathrm{E}_{\tilde{q}_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\widetilde{\mathbf{x}})}\right],$$

where $\tilde{q}_\phi(\mathbf{z}|\mathbf{x}) = \int q_\phi(\mathbf{z}|\tilde{\mathbf{x}}) p(\tilde{\mathbf{x}}|\mathbf{x}) \mathrm{d}\tilde{\mathbf{x}}$.

However, the noise in this case was set to a constant during the training procedure. To the best of our knowledge no one analysed how the scheduling scheme might influence the learning of the auto-encoder's structure as well as the approximate form of the posterior of the latent variable. We believe that combination of both scheduled denoising training with the variational form of an auto-encoder should lead to gains in terms of the optimising lower bound and improving the reconstruction error as it was the case in the section 5.

8. Conclusions

The analysed paper provides a clear introduction to a new methodology for performing variational inference. We managed to obtain the same results as the authors on both data sets which was partially possible thanks to the straightforward explanation of the proposed method and the detailed description of the experiments conducted. We found the model structure very robust to changes of parameters of the network. Moreover, our experiments show that the performance of VAEB is superior to that of the vanilla AE architecture and it is resistant to superfluous latent variables thanks to automatic regularisation via the KL-term. Our implementation of variational inference on both the parameters θ and the latent variables $\mathbf{z}$ performed disappointingly poorly which might be partially explained by overly-restrictive prior and we plan to further investigate this problem. The code needed to replicate our results is available at `https://github.com/budzianowski/VAEB`. We found this project incredibly interesting and we hope to undertake further work on the extensions proposed in the previous section.

References

[1] Charles Blundell et al. "Weight uncertainty in neural networks". In: *arXiv preprint arXiv:1505.05424* (2015).

[2] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive subgradient methods for online learning and stochastic optimization". In: *The Journal of Machine Learning Research* 12 (2011), pp. 2121–2159.

[3] Krzysztof J. Geras and Charles Sutton. "Scheduled denoising autoencoders". In: *arXiv preprint arXiv:1406.3269* (2014).

[4] Daniel Jiwoong Im et al. "Denoising Criterion for Variational Auto-Encoding Framework". In: *arXiv preprint arXiv:1511.06406* (2015).

[5] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[6] Dougal Maclaurin, David Duvenaud, and Ryan P. Adams. "Autograd: Effortless Gradients in Numpy". In: ().

[7] Michalis K Titsias. "Variational learning of inducing variables in sparse Gaussian processes". In: *International Conference on Artificial Intelligence and Statistics*. 2009, pp. 567–574.

[8] Dustin Tran, Rajesh Ranganath, and David M. Blei. "Variational Gaussian Process". In: *arXiv preprint arXiv:1511.06499* (2015).

[9] Pascal Vincent et al. "Extracting and composing robust features with denoising autoencoders". In: (2008), pp. 1096–1103.