WiSE-ALE: Wide Sample Estimator for Approximate Latent Embedding

Anonymous authors
Paper under double-blind review

Abstract

Variational Auto-encoders (VAEs) have been very successful as methods for forming compressed latent representations of complex, often high-dimensional, data. In this paper, we derive an alternative variational lower bound from the one common in VAEs, which aims to minimize aggregate information loss. Using our lower bound as the objective function for an auto-encoder enables us to place a prior on the *bulk statistics*, corresponding to an aggregate posterior for the entire dataset, as opposed to a single sample posterior as in the original VAE. This alternative form of prior constraint allows individual posteriors more flexibility to preserve necessary information for good reconstruction quality. We further derive an analytic approximation to our lower bound, leading to an efficient learning algorithm - WiSE-ALE. Through various examples, we demonstrate that WiSE-ALE can reach excellent reconstruction quality in comparison to other state-of-the-art VAE models, while still retaining the ability to learn a smooth, compact representation.

1 Introduction

Unsupervised learning is a central task in machine learning. Its objective can be informally described as learning a representation of some observed forms of information in a way that the representation summarizes the overall statistical regularities of the data (Barlow, 1989). Deep generative models are a popular choice for unsupervised learning, as they marry deep learning with probabilistic models to estimate a joint probability between high dimensional input variables $\mathbf{x}$ and unobserved latent variables $\mathbf{z}$. Early successes of deep generative models came from Restricted Boltzmann Machines (Hinton & Salakhutdinov, 2006) and Deep Boltzmann Machines (Salakhutdinov & Hinton, 2009), which aim to learn a compact representation of data. However, the fully stochastic nature of the network requires layer-by-layer pre-training using MCMC-based sampling algorithms, resulting in heavy computation cost.

Kingma & Welling (2013) consider the objective of optimizing the parameters in an auto-encoder network by deriving an analytic solution to a variational lower bound of the log likelihood of the data, leading to the Auto-Encoding Variational Bayes (AEVB) algorithm. They apply a reparameterization trick to maximally utilize deterministic mappings in the network, significantly simplifying the training procedure and reducing instability. Furthermore, a regularization term naturally occurs in their model, allowing a prior $p(\mathbf{z})$ to be placed over every sample embedding $q(\mathbf{z}|\mathbf{x})$. As a result, the learned representation becomes compact and smooth; see e.g. Fig. 1 where we learn a 2D embedding of MNIST digits using 4 different methods and visualize the aggregate posterior distribution of 64 random samples in the learnt 2D embedding space.

However, because the choice of the prior is often uninformative, the smoothness constraint imposed by this regularization term can cause information loss between the input samples and the latent embeddings, as shown by the merging of individual embedding distributions in Fig. 1(d) (especially in the outer areas away from zero code). Extreme effects of such behaviours can be noticed from β-VAE (Higgins et al., 2016), a derivative algorithm of AEVB which further increases the weighting on the regularizing term with the aim of learning an even smoother, disentangled representation of the data. As shown in Fig. 1(e), the individual embedding distributions are almost indistinguishable, leading to an overly severe information bottleneck which can cause high rates of distortion (Tishby et al., 1999). The other end of the spectrum can be indicated by Fig. 1(b), where perfect reconstruction

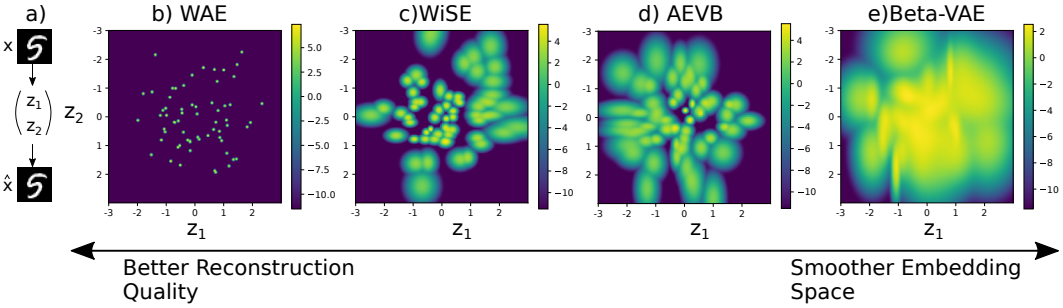


Figure 1: (a) Learning a 2D embedding of MNIST handwritten digits through an auto-encoding framework. Embedding distributions (aggregate posteriors) of 64 randomly drawn MNIST digits when WAE (b), our proposed WiSE-ALE (c), AEVB (d) or β-VAE (e) is used for the learning. Different learning algorithms find a different level of tradeoff between the reconstruction quality (information preservation) and the smoothness of the posterior distribution.

can be achieved but the learnt embedding distributions appear to severely sharp, indicating a latent representation which is heavily non-smooth and likely to be unstable due to a small amount of noise.

In this paper, we propose WiSE-ALE (a wide sample estimator), which imposes a prior on the *bulk statistics* of a mini-batch of latent embeddings. Learning under our WiSE-ALE objective does not penalize individual embeddings lying away from the zero code, so long as the aggregate distribution (the average of all individual embedding distributions) does not violate the prior significantly. Hence, our approach mitigates the distortion caused by the current form of the prior constraint in the AEVB objective. Furthermore, the objective of our WiSE-ALE algorithm is derived by applying variational inference in a simple latent variable model (Section 2) and with further approximation, we derive an analytic form of the learning objective, resulting in efficient learning algorithm.

In general, the latent representation learned using our algorithm enjoys the following properties: 1) **smoothness**, as indicated in Fig. 1(d), the probability density for each individual embedding distribution decays smoothly from the peak value; 2) **compactness**, as individual embeddings tend to occupy a maximal local area in the latent space with minimal gaps in between; and 3) **separation**, indicated by the narrow, but clear borders between neighbouring embedding distributions as opposed to the merging seen in AEVB. In summary, our contributions are:

- **An alternative variational lower bound** to the data log likelihood is derived, allowing us to impose prior constraint on the *bulk statistics* of a mini-batch embedding distributions.
- **Analytic approximations** to the lower bound are derived, allowing efficient optimization without sampling-based methods and leading to our WiSE-ALE algorithm.
- **Extensive analysis** of our algorithm's performance in comparison with three related VAE algorithms, namely AEVB, β-VAE and WAE (Tolstikhin et al., 2017).

In the rest of the paper, we first review directed graphical models in Section 2. We then derive our variational lower bound and its analytic approximations in Section 3. Related work is discussed in Section 4. Experiment results are analyzed in Section 5, leading to conclusions in Section 6.

2 Background: Latent Variable Models

Here we briefly review the latent variable model that allows variational inference through an auto-encoding task and highlight the difference between the latent variable model for our WiSE-ALE algorithm and that for the AEVB algorithm (Kingma & Welling, 2013).

Given N observations of input samples $\boldsymbol{x} \in R^{d_x}$ denoted $\mathcal{D}_N = \left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(N)}\right)$, we assume $\mathbf{x}$ is generated from a latent variable $\mathbf{z} \in R^{d_z}$ of a much lower dimension. Here we denote $\mathbf{x}$ and $\mathbf{z}$ as random variables, $\boldsymbol{x}^{(i)}$ or $\boldsymbol{z}^{(i)}$ as the i-th input or latent code sample (i.e. a vector), and $\mathbf{x}_i$ and $\mathbf{z}_i$ as the random variable for $\boldsymbol{x}^{(i)}$ and $\boldsymbol{z}^{(i)}$. As shown in Fig. 2(a), this generative process can be modelled by a simple directed graphical model (Jordan et al., 1999), which models the

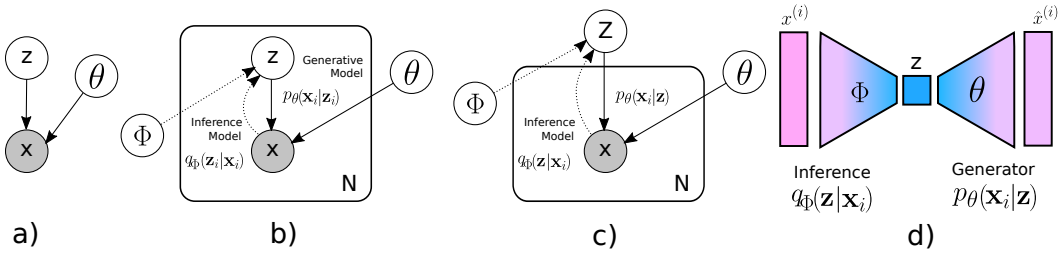


Figure 2: (a) Directed graphical models for the generative model between observation $\mathbf{x}$ and latent variable $\mathbf{z}$. (b) Latent variable model for the AEVB algorithm, where z indicates N random variables for N latent codes. (c) Latent variable model for the our WiSE-ALE algorithm, where z is a single random variable for the aggregate posterior of the entire dataset. (d) A generic neural network implementation suitable for both (b) and (c).

joint probability distribution $p_\theta(\mathbf{x}, \mathbf{z}|\mathcal{D}_N) = p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathcal{D}_N) = p_\theta(\mathbf{z}|\mathbf{x})p(\mathbf{x}|\mathcal{D}_N)$ between $\mathbf{x}$ and $\mathbf{z}$, given the current observations $\mathcal{D}_N$. $p(\mathbf{z}|\mathcal{D}_N)$ is the latent distribution given $\mathcal{D}_N$, $p(\mathbf{x}|\mathcal{D}_N)$ is the data distribution for $\mathcal{D}_N$ and $p_\theta(\mathbf{x}|\mathbf{z})$ and $p_\theta(\mathbf{z}|\mathbf{x})$ denote the complex transformation from the latent to the input space and reverse, where the transformation mapping is parameterised by θ. The learning task is to estimate the optimal set of $\boldsymbol{\theta}$ so that this latent variable model can explain the data $\mathcal{D}_N$ well.

As the inference of the latent variable $\mathbf{z}$ given $\mathbf{x}$ (i.e. $p_\theta(\mathbf{z}|\mathbf{x})$) cannot be directly estimated because $p(\mathbf{x}|\mathcal{D}_N)$ is unknown, both AEVB (Fig. 2(b)) and our WiSE-ALE (Fig. 2(c)) resort to variational method to approximate the target distribution $p_\theta(\mathbf{z}|\mathbf{x})$ by a proposal distribution $q_\phi(\mathbf{z}|\mathbf{x})$ with the modified learning objective that both θ and ϕ are optimised so that the model can explain the data well and $q_\phi(\mathbf{z}|\mathbf{x})$ approaches $p_\theta(\mathbf{z}|\mathbf{x})$. The primary difference between the AEVB model and our WiSE-ALE model lies in how the joint probability $p_\theta(\mathbf{x}, \mathbf{z}|\mathcal{D}_N)$ is modelled and specifically whether we assume an individual random variable for each latent code $\boldsymbol{z}^{(i)}$. The AEVB model assumes a pair of random variables $(\mathbf{x}_i, \mathbf{z}_i)$ for each $\boldsymbol{x}^{(i)}$ and estimates the joint probability as

$$
\begin{aligned}
p_\theta(\mathbf{x}, \mathbf{z}|\mathcal{D}_N) &= p_\theta(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N|\, \mathcal{D}_N) && (1)\\
&= p_\theta(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N|\, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N)\; p_\theta(\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N|\, \mathcal{D}_N) && (2)\\
&= \prod_{i=1}^{N} p_\theta(\mathbf{x}_i|\, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_N) \prod_{i=1}^{N} p_\theta(\mathbf{z}_i|\, \mathcal{D}_N) && (3)\\
&= \prod_{i=1}^{N} p_\theta(\mathbf{x}_i|\mathbf{z}_i) \prod_{i=1}^{N} p_\theta(\mathbf{z}_i|\, \mathcal{D}_N) && (4)\\
&= \prod_{i=1}^{N} \Big(p_\theta(\mathbf{x}_i|\mathbf{z}_i)\; p_\theta(\mathbf{z}_i|\mathcal{D}_N) \Big). && (5)
\end{aligned}
$$

The equality between Eq. 2 and Eq. 3 can only be made with the assumption that the generation process for each $\mathbf{x}_i$ is independent (first product in Eq. 3) and each $\mathbf{z}_i$ is also independent (second product in Eq. 3). Such interpretation of the joint probability leads to the latent variable model in Fig. 2(b) and the prior constraint (often taken as $\mathcal{N}(0, \boldsymbol{I})$ to encourage shrinkage when no data is observed) is imposed on every $\mathbf{z}_i$.

In contrast, our WiSE-ALE model takes a single random variable to estimate the latent distribution for the entire dataset $\mathcal{D}_N$. Hence, the joint probability in our model can be broken down as

$$
\begin{aligned}
p_\theta(\mathbf{x}, \mathbf{z}|\, \mathcal{D}_N) &= p_\theta(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N, \mathbf{z}|\, \mathcal{D}_N) && (6)\\
&= p_\theta(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N, |\, \mathbf{z})\; p_\theta(\mathbf{z}|\, \mathcal{D}_N) && (7)\\
&= p_\theta(\mathbf{z}|\mathcal{D}_N) \prod_{i=1}^{N} p_\theta(\mathbf{x}_i|\, \mathbf{z}), && (8)
\end{aligned}
$$

leading to the latent variable model illustrated in Fig. 2(c). The only assumption we make in our model is assuming the generative process of different input samples given the latent distribution of

the current dataset as independent, which we consider as a sensible assumption. More significantly, we do not require independence between different $\mathbf{z}_i$ as opposed to the AEVB model, leading to a more flexible model. Furthermore, the prior constraint in our model is naturally imposed on the aggregate posterior $p(\mathbf{z}|\mathcal{D}_N)$ for the entire dataset, leading to more flexibility for each individual sample latent code to shape an embedding distribution to preserve a better quality of information about the corresponding input sample.

Neural networks can be used to parameterize $p_\theta(\mathbf{x}_i|\mathbf{z}_i)$ in the generative model and $q_\phi(\mathbf{z}_i|\mathbf{x}_i)$ in the inference model from the AEVB latent variable model or $p_\theta(\mathbf{x}_i|\mathbf{z})$ and $q_\phi(\mathbf{z}|\mathbf{x}_i)$ correspondingly from our WiSE-ALE latent variable model. Both networks can be implemented through an auto-encoder network illustrated in Fig. 2(d).

3 Our Method

In this section, we first define the aggregate posterior distribution $p(\mathbf{z}|\mathcal{D}_N)$ which serves as a core concept in our WiSE-ALE proposal. We then derive a variational lower bound to the marginal log likelihood of the data $\log p(\mathcal{D}_N)$ with the focus on the aggregate posterior distribution. Further, analytic approximation to the lower bound is derived, allowing efficient optimization of the model parameters and leading to our WiSE-ALE learning algorithm. Intuition of our proposal is also discussed.

3.1 Aggregate Posterior

Here we formally define the aggregate posterior distribution $p(\mathbf{z}|\mathcal{D}_N)$, i.e. the latent distribution given the entire dataset $\mathcal{D}_N$. Considering

$$p(\mathbf{z}|\mathcal{D}_N) = \int p_\theta(\mathbf{z}|\mathbf{x}_j)p(\mathbf{x}_j|\mathcal{D}_N)\mathrm{d}\mathbf{x}_j = \sum_{i=1}^{N} p_\theta(\mathbf{z}|\mathbf{x}_j = \boldsymbol{x}^{(i)})P(\mathbf{x}_j = \boldsymbol{x}^{(i)}|\mathcal{D}_N) = \frac{1}{N}\sum_{i=1}^{N} p_\theta(\mathbf{z}|\boldsymbol{x}^{(i)}), \tag{9}$$

we have the aggregate posterior distribution for the entire dataset as the average of all the individual sample posteriors. The second equality in Eq. 9 is made approximating the integral through summation. The third equality is obtained following the conventional assumption in the VAE literature that each input sample, $\boldsymbol{x}^{(i)}$, is drawn from the data set $\mathcal{D}_N$ with equal probability, i.e. $P(\boldsymbol{x}^{(i)}|\mathcal{D}_N) = \frac{1}{N}$. Similarly, for the estimated aggregate posterior distribution $q(\mathbf{z}|\mathcal{D}_N)$, we have

$$q(\mathbf{z}|\mathcal{D}_N) = \frac{1}{N}\sum_{i=1}^{N} q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)}). \tag{10}$$

3.2 Alternative Variational Lower Bound (LB)

To carry out variational inference, we minimize the KL divergence between the estimated and the true aggregate posterior distributions $q_\phi(\mathbf{z}|\mathcal{D}_N)$ and $p_\theta(\mathbf{z}|\mathcal{D}_N)$, i.e.

$$D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p_\theta(\mathbf{z}|\mathcal{D}_N)\big] \;=\; \mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}\left[\log \frac{q_\phi(\mathbf{z}|\mathcal{D}_N)}{p_\theta(\mathbf{z}|\mathcal{D}_N)}\right]. \tag{11}$$

Substituting $p_\theta(\mathbf{z}|\mathcal{D}_N) = \frac{p_\theta(\mathcal{D}_N|\mathbf{z})\,p(\mathbf{z})}{p_\theta(\mathcal{D}_N)}$ in Eq. 11 and breaking down the products and fractions inside the log, we have

$$D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p_\theta(\mathbf{z}|\mathcal{D}_N)\big] \;=\; \mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}\big[\log\, q_\phi(\mathbf{z}|\mathcal{D}_N) - \log\, p_\theta(\mathcal{D}_N|\mathbf{z}) - \log p(\mathbf{z})\big] \;+\; \log\, p(\mathcal{D}_N).$$

Re-arranging the above equation, we have

$$\log\, p(\mathcal{D}_N) \;-\; D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p_\phi(\mathbf{z}|\mathcal{D}_N)\big] \;=\; \mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}\big[\log p_\theta(\mathcal{D}_N|\mathbf{z})\big] - D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big].$$

As $D_{\mathrm{KL}}\big[q_\phi(\mathbf{z}|\mathcal{D}_N)\|p_\phi(\mathbf{z}|\mathcal{D}_N)\big]$ is non-negative, we have obtained a variational lower bound $L^{\text{WiSE-ALE}}(\phi,\theta;\mathcal{D}_N)$ to the marginal log likelihood of the data $\log p(\mathcal{D}_N)$ as

$$\log p(\mathcal{D}_N) \;\geq\; L^{\text{WiSE-ALE}}(\phi,\theta;\mathcal{D}_N) \;=\; \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}\big[\log p_\theta(\mathcal{D}_N|\mathbf{z})\big]}_{\text{(1) Reconstruction likelihood}} \;-\; \underbrace{D_{\mathrm{KL}}\big[\,q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big]}_{\text{(2) Prior constraint}}. \tag{12}$$

There are two terms in the derived lower bound: (1) a **reconstruction likelihood** term that indicates how likely the current dataset $\mathcal{D}_N$ are generated by the aggregate latent posterior distribution $q_\phi(\mathbf{z}|\mathcal{D}_N)$ and (2) a **prior constraint** that penalizes severe deviation of the aggregate latent posterior distribution $q_\phi(\mathbf{z}|\mathcal{D}_N)$ from the preferred prior $p(\mathbf{z})$, acting naturally as a regularizer. By maximizing the lower bound $L^{\text{WiSE-ALE}}(\phi,\theta;\mathcal{D}_N)$ defined in Eq. 12, we are approaching to $\log p(\mathcal{D}_N)$ and, hence, obtaining a set of parameters θ and ϕ that find a natural balance between a good reconstruction likelihood (good explanation of the observed data) and a reasonable level of compliance to the prior assumption (achieving some preferable properties of the posterior distribution, such as smoothness and compactness).

3.3 Approximation of the Proposed Lower Bound

To allow fast and efficient optimization of the model parameters θ and ϕ, we derive analytic approximations for the two terms in our proposed lower bound (Eq. 12).

3.3.1 Approximation to Reconstruction Likelihood Term

To approximate (1) **reconstruction likelihood** term in Eq. 12, we first substitute the definition of the approximate aggregate posterior given in Eq. 10 in the expectation operation in $\mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}\big[\log p_\theta(\mathcal{D}_N|\mathbf{z})\big]$, i.e.

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}\big[\log p_\theta(\mathcal{D}_N|\mathbf{z})\big] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p_\theta(\mathcal{D}_N|\mathbf{z})\big]. \tag{13}$$

Now we can decompose the $p_\theta(\mathcal{D}_N|\mathbf{z})$ as a product of individual sample likelihood, due to the conditional independence, i.e.

$$\log p_\theta(\mathcal{D}_N|\mathbf{z}) = \log\prod_{j=1}^{N} p_\theta(\boldsymbol{x}^{(j)}|\mathbf{z}) = \sum_{j=1}^{N}\log p_\theta(\boldsymbol{x}^{(j)}|\mathbf{z}). \tag{14}$$

Substituting this into Eq. 13, we have

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}[\log p_\theta(\mathcal{D}_N|\mathbf{z})] = \sum_{i=1}^{N}\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\left[\frac{1}{N}\sum_{j=1}^{N}\log p_\theta(\boldsymbol{x}^{(j)}|\mathbf{z})\right]. \tag{15}$$

Eq. 15 can be used to evaluate the reconstruction likelihood for $\mathcal{D}_N$. However, learning directly with this reconstruction estimate does not lead to convergence in our experiments and the computation is quite costly, as for every evaluation of the reconstruction likelihood, we need to evaluate N^2 expectation operations. We choose to simplify the reconstruction likelihood further to be able to reach convergence during learning at the cost of losing the lower bound property of the objective function $L^{\text{WiSE-ALE}}(\phi,\theta;\mathcal{D}_N)$. Firstly, we apply Jensen inequality to the term inside the expectation in Eq. 15, leading to an upper bound of the reconstruction likelihood term as

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}[\log p_\theta(\mathcal{D}_N|\mathbf{z})] \leq \sum_{i=1}^{N}\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\left[\log\left(\frac{1}{N}\sum_{j=1}^{N}p_\theta(\boldsymbol{x}^{(j)}|\mathbf{z})\right)\right]. \tag{16}$$

Now $(N-1)$ sample-WiSE-ALE likelihood distributions in the summation inside the $\log$ can be dropped with the assumption that the $p_\theta(\boldsymbol{x}^{(j)}|\mathbf{z})$ will only be non-zero if $\mathbf{z}$ is sampled from the

posterior distribution of the same sample $\boldsymbol{x}^{(j)}$ at the encoder, i.e. $i = j$. Therefore, the approximation becomes

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathcal{D}_N)}[\log p_\theta(\mathcal{D}_N|\mathbf{z})] \leq \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\Big[\log p_\theta(\boldsymbol{x}^{(i)}|\mathbf{z})\Big] - N\log N. \tag{17}$$

Using the approximation of the reconstruction likelihood term given by Eq. 17 rather than Eq. 15, we are able to reach convergence efficiently during learning at the cost of the estimated objective no longer remaining a lower bound to $\log p(\mathcal{D}_N)$. Details of deriving the above approximation are given in Appendix A.

3.3.2 Approximation to Prior Constraint Term

The (2) **prior constraint** term $D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big]$ in our objective function (Eq. 12) evaluates the KL divergence between the approximate aggregate posterior distribution $q_\phi(\mathbf{z}|\mathcal{D}_N)$ and a zero-mean, unit-variance Gaussian distribution $p(\mathbf{z})$. Here we assume that each sample-WiSE-ALE posterior distribution can be modelled by a factorial Gaussian distribution, i.e. $q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)}) = \prod_{k=1}^{d_z}\mathcal{N}\big(\mathbf{z}_k|\,\mu_k(\boldsymbol{x}^{(i)}), \sigma_k^2(\boldsymbol{x}^{(i)})\big)$, where k indicates the k-th dimension of the latent variable $\mathbf{z}$ and $\mu_k(\boldsymbol{x}^{(i)})$ and $\sigma_k^2(\boldsymbol{x}^{(i)})$ are the mean and variance of the k-th dimension embedding distribution for the input $\boldsymbol{x}^{(i)}$. Therefore, $D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big]$ computes the KL divergence between a mixture of Gaussians (as Eq. 10) and $\mathcal{N}(0, \boldsymbol{I})$. There is no analytical solution for such KL divergences. Hence, we derive an analytic upper bound allowing for efficient evaluation.

Firstly, we substitute $q_\phi(\mathbf{z}|\mathcal{D}_N) = \frac{1}{N}\sum_{i=1}^{N} q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})$ (Eq. 10) to $D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big]$, giving

$$D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big] = \frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log q_\phi(\mathbf{z}|\mathcal{D}_N)\big] - \mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big]\Big). \tag{18}$$

Applying Jensen inequality, i.e. $\mathbb{E}_x\big[\log f(x)\big] \leq \log \mathbb{E}_x\big[f(x)\big]$, to the first term inside the summation in Eq. 18, we have

$$D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big] \;\leq\; \mathrm{KL}_{\text{approx}}^{\text{UB}} \tag{19}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\Big(\log \mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[q_\phi(\mathbf{z}|\mathcal{D}_N)\big]\Big) \;-\; \frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big]\Big). \tag{20}$$

Taking advantage of the Gaussian assumption for $q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})$ and $p(\mathbf{z})$, we can compute the expectations in Eq. 20 analytically with the result quoted below and the full derivation given in Appendix B.1.

$$\mathrm{KL}_{\text{approx}}^{\text{UB}} \;=\; \frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{N}\sum_{j=1}^{N}\prod_{k=1}^{d_z} A^{-1/2}B\right) + \frac{1}{2N}\sum_{i=1}^{N}\sum_{k=1}^{d_z} C, \tag{21}$$

$$\text{where}\;\; A \;=\; 2\pi\big((\sigma_k^{(i)})^2 + (\sigma_k^{(j)})^2\big), \tag{22}$$

$$B \;=\; \exp\left(-\frac{1}{2}\frac{\big(\mu_k^{(i)} - \mu_k^{(j)}\big)^2}{(\sigma_k^{(i)})^2 + (\sigma_k^{(j)})^2}\right), \tag{23}$$

$$C \;=\; (\sigma_k^{(i)})^2 + (\mu_k^{(i)})^2 + \log 2\pi. \tag{24}$$

When the overall objective function $L^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{D}_N)$ in Eq. 12 is maximised, this upper bound approximation will approach the true KL divergence $D_{\mathrm{KL}}\big[\, q_\phi(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big]$, which ensures that the prior constraint on the overall aggregate posterior distribution takes effects.

3.3.3 Overall Objective Functions

Combining results from Section 3.3.1 and 3.3.2, we obtain an analytic approximation $L_{\text{approx}}^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{D}_N)$ for the variational lower bound $L^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{D}_N)$ defined in Eq. 12, as

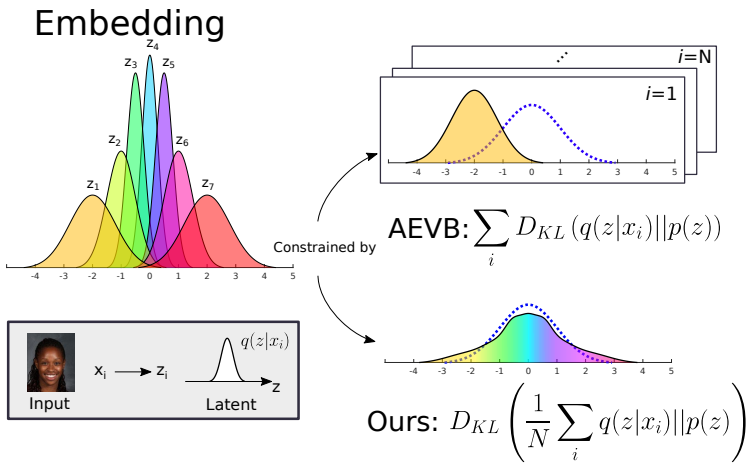


Figure 3: Comparison between our WiSE-ALE learning scheme and the AEVB estimator. AEVB imposes the prior constraint on every sample embedding distribution, whereas our WiSE-ALE imposes the constraint to the overall aggregate embedding distribution over the entire dataset (over a mini-batch as an approximation for efficient learning).

shown below:

$$L_{\text{approx}}^{\text{WiSE-ALE}}(\phi,\theta;\mathcal{D}_N) \;=\; \sum_{i=1}^{N}\mathcal{L}(\phi,\theta|\boldsymbol{x}^{(i)}) \;-\; KL\big[\,q_\phi(\mathbf{z}|\mathcal{D}_N)\,||\,p(\mathbf{z})\big], \tag{25}$$

where we use $\mathcal{L}\big(\phi,\theta\,|\,\boldsymbol{x}^{(i)}\big)$ to denote the sample-WiSE-ALE reconstruction likelihood $\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})}\Big[\log p_\theta(\boldsymbol{x}^{(i)}|\mathbf{z})\Big]$ given by Eq. 17 and the KL divergence term is estimated through $\text{KL}_{\text{approx}}^{\text{UB}}$ defined in Eq. 21. Optimizing $L_{\text{approx}}^{\text{WiSE-ALE}}(\phi,\theta;\mathcal{D}_N)$ w.r.t the model parameters ϕ and θ, we are able to learn a model that naturally balances between a good embedding of the observed data and some preferred properties of the latent embedding distributions, such as smoothness and compactness.

3.4 Comparison of AEVB and WiSE-ALE Learning Objective Functions

Comparing the objective function in our WiSE-ALE algorithm and that proposed in AEVB algorithm (Kingma & Welling, 2013) stated below,

$$L^{\text{AEVB}}(\phi,\theta;\mathcal{D}_N) \;=\; \sum_{i=1}^{N}\mathcal{L}(\phi,\theta|\boldsymbol{x}^{(i)}) \;-\; \sum_{i=1}^{N}D_{\text{KL}}\big[\,q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})\|p(\mathbf{z})\big]. \tag{26}$$

we notice that the difference lies in the form of prior constraint and the difference is illustrated in Fig. 3. AEVB learning algorithm imposes the prior constraint on every sample embedding and any deviation away from the zero code or the unit variance (e.g. variance of a sample posterior becomes less than 1, as the model becomes more certain about a specific input sample embedding) will incur penalty. In contrast, our WiSE-ALE learning objective imposes the prior constraint on the aggregate posterior distribution, i.e. the average of all the sample embeddings. Such prior constraint will allow more flexibility for each sample posterior to settle at a mean and variance value in favour for good reconstruction quality, while preventing too large mean values (acting as a regulariser) or too small variance values (ensuring smoothness of the learnt latent representation).

To investigate the different behaviours of the two prior constraints more concretely, we consider only two embedding distributions $q(\mathbf{z}|\boldsymbol{x}^{(1)})$ and $q(\mathbf{z}|\boldsymbol{x}^{(2)})$ (red dashed lines) in a 1D latent space, as shown in Fig. 4. The mean values of the two embedding distributions are fixed to make the analysis simple and their variances are allowed to change. When the variances of the two embedding distributions are large, such as Fig. 4(a), $q(\mathbf{z}|\boldsymbol{x}^{(1)})$ and $q(\mathbf{z}|\boldsymbol{x}^{(2)})$ have a large area of overlap and it is difficult to distinguish the input samples $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ in the latent space. On the other hand, when the two

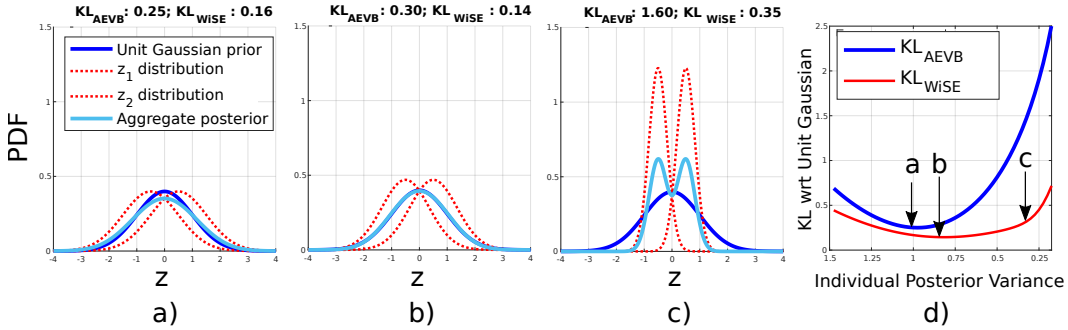


Figure 4: Comparison of the prior constraint in our objective function and that in the AEVB objective function. In a-c, red dashed lines are two sample-WiSE-ALE posterior distributions $q(\mathbf{z}|\boldsymbol{x}^{(1)})$ and $q(\mathbf{z}|\boldsymbol{x}^{(2)})$ which embed the inputs $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ in the latent space (the more separable $q(\mathbf{z}|\boldsymbol{x}^{(1)})$ and $q(\mathbf{z}|\boldsymbol{x}^{(2)})$, the easier to distinguish $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ in the latent space), dark blue line is $\mathcal{N}(0, \boldsymbol{I})$ prior distribution, light blue line is the aggregate posterior (average of the two individual posteriors). The posteriors given by (a) the minimal KL value in AEVB objective, (b) the minimal KL value in our WiSE-ALE objective and (c) an acceptable KL value in our WiSE-ALE objective. (d) comparison of KL values in the AEVB and our WiSE-ALE objectives across different posterior variances.

embedding distributions have small variances, such as Fig. 4(c), there is clear separation between $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ in the latent space, indicating the embedding only introduces a small level of information loss. Overall, the prior constraint in the AEVB objective favours the embedding distributions much closer to the uninformative $\mathcal{N}(0, \boldsymbol{I})$ prior, leading to large area of overlap between the individual posteriors, whereas our WiSE-ALE objective allows a wide range of acceptable embedding mean and variance, which will then offer more flexibility in the learnt posteriors to maintain a good reconstruction quality.

3.5 Efficient Mini-Batch Estimator

So far our derivation has been for the entire dataset $\mathcal{D}_N$. Given a small subset $\mathcal{B}_M$ with M samples randomly drawn from $\mathcal{D}_N$, we can obtain a variational lower bound for a mini-batch as:

$$L^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{B}_M) = \sum_{i=1}^{M} \left(\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})} \left[\log p_\theta(\boldsymbol{x}^{(i)}|\mathbf{z}) \right] \right) - D_{\text{KL}} \left[\, q_\phi(\mathbf{z}|\mathcal{B}_M) \| p(\mathbf{z}) \right]. \tag{27}$$

When $\mathcal{B}_M$ is reasonably large, then $L^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{B}_M)$ becomes an good approximation of $L^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{D}_N)$ through

$$L^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{D}_N) \approx \frac{N}{M} L^{\text{WiSE-ALE}}(\phi, \theta; \mathcal{B}_M). \tag{28}$$

Given the expressions for the objective functions derived in Section 3.3, we can compute the gradient for an approximation to the lower bound of a mini-batch $\mathcal{B}_M$ and apply stochastic gradient ascent algorithm to iteratively optimize the parameters ϕ and θ. We can thus apply our WiSE-ALE algorithm efficiently to a mini-batch and learn a meaningful internal representation of the entire dataset. Algorithmically, WiSE-ALE is similar to AEVB, save for an alternate objective function as per Section 3.3.3. The procedural details of the algorithm are presented in Appendix C.

4 Related Work

Bengio et al. (2013) proposes that a learned representation of data should exhibit some general features, such as smoothness, sparsity and simplicity. These attributes are general, however, and are not tailored to any specific downstream tasks. Requirements from Bayesian decision making (see e.g. Lacoste-Julien et al. (2011); Cobb et al. (2018)) adds consideration of a target task and proposes latent distribution approximations which optimise the performance over a particular task, as well as conforming to more general properties. The AEVB algorithm (Kingma & Welling, 2013) learns the latent posterior distribution under a reconstruction task, while simultaneously satisfying

the prior, thus ensuring that the representation is smooth and compact. However, the prior form of the AEVB algorithm imposes significant influence on the solution space (as discussed in Section 3.4), and leads to a sacrifice of reconstruction quality. Our WiSE-ALE algorithm, however, prioritises the reconstruction task yet still enables globally desirable properties.

WiSE-ALE is, however, not the only algorithm that considers an alternate prior form to mitigate its impact on reconstruction quality. The Gaussian Mixture VAE (Dilokthanakul et al., 2016) uses a Gaussian mixture model to parameterise $p(\mathbf{z})$, encouraging more flexible sample posteriors. The Adversarial Auto-Encoder (Makhzani et al., 2016) matches the aggregate posterior over the latent variables with a prior distribution through adversarial training. The WAE (Tolstikhin et al., 2017) minimises a penalised form of the Wasserstein distance between the aggregate posterior distribution and the prior, claiming a generalisation of the AAE algorithm under the theory of optimal transport (Villani, 2008). More recently, the Sinkhorn Auto-Encoder (Patrini et al., 2018) builds a formal analysis of auto-encoders using an optimal transport based prior and uses the Sinkhorn algorithm as an alternative to estimate the Wasserstein distance in WAE.

Our work differs from these in two main aspects. Firstly, our objective function can be evaluated analytically, leading to an efficient optimization process. In many of the above work, the optimization involves adversarial training and some hyper-parameter tuning, which leading to less efficient learning and slow or even no convergence. Secondly, our WiSE-ALE algorithm naturally finds a balance between good reconstruction quality and preferred latent representation properties, such as smoothness and compactness, as shown in Fig. 1(c). In contrast, some other work sacrifice the properties of smoothness and compactness severely for improved reconstruction quality, as shown in Fig. 1(b). Many works (Bloesch et al., 2018; Clark et al., 2018) have indicated that those properties of the learnt latent representation are essential for tasks that require optimisation over the latent space.

5 Experiments

We evaluate our WiSE-ALE algorithm in comparison with AEVB, β-VAE and WAE on the following 3 datasets. The implementation details for all experiments are given in Appendix E.

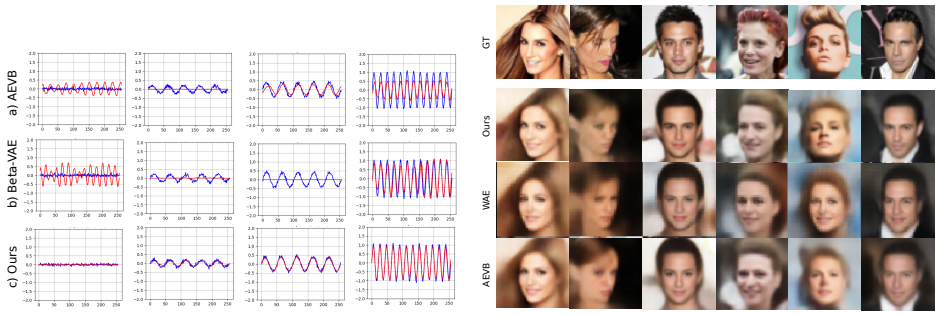
1. **Sine Wave**. We generated 200,000 sine waves with small random noise: $x(t) = A\sin(2\pi ft + \varphi) + \epsilon$, each containing 256 samples, with independently sampled frequency $f \sim \text{Unif}(0, 20\text{Hz})$, phase angle $\varphi \sim \text{Unif}(0, 2\pi)$ and amplitude $A \sim \text{Unif}(0, 2)$.
2. **MNIST** (LeCun, 1998). 70,000 28×28 binary images that contain hand-written digits.
3. **CelebA** (Liu et al., 2015). 202,599 RGB images of aligned celebrity faces of 218×178 are cropped to square images of 178×178 and resized to 64×64.

5.1 Reconstruction Quality

Throughout all experiments, our method has shown consistently superior reconstruction quality compared to AEVB, β-VAE and WAE. Fig. 5 offers a graphical comparison across the reconstructed samples given by different methods for the sine wave and CelebA datasets. For the sine wave dataset, our WiSE-ALE algorithms achieves almost perfect reconstruction, whereas AEVB and β-VAE often struggle with low-frequency signals and have difficulty predicting the amplitude correctly. For the CelebA dataset, our WiSE-ALE manages to predict much sharper human faces, whereas the AEVB predictions are often blurry and personal characteristics are often ignored. WAE reaches a similar level of reconstruction quality to ours in some images, but it sometimes struggles with discovering the right elevation and azimuth angles, as shown in the second to the right column in Fig. 5b.

5.2 Properties of the Learnt Representation Space

We understand that a good latent representation should not only reconstruct well, but also preserve some preferable qualities, such as smoothness, compactness and possibly meaningful interpretation of the original data. Fig. 1 indicates that our WiSE-ALE automatically learns a latent representation that finds a good tradeoff between minimizing the information loss and maintaining a smooth and compact aggregate posterior distribution. Furthermore, as shown in Fig. 6, we compare the ELBO values given by AEVB, β-VAE and our WiSE-ALE over training for the Sine dataset. Our WiSE-ALE manages to



(a) Reconstructed sine waves given by AEVB, β-VAE and our WiSE-ALE.

(b) Reconstructed celebrity faces given by AEVB, WAE and our WiSE-ALE (CelebA dataset).

Figure 5: Qualitative comparison of the reconstruction quality between our WiSE-ALE and other methods.

report the highest ELBO with a significantly lower reconstruction error and a fairly good performance in the KL divergence loss. This indicates that our WiSE-ALE is able to learn an overall good quality representation that is closest to the true latent distribution which gives rise to the data observation.

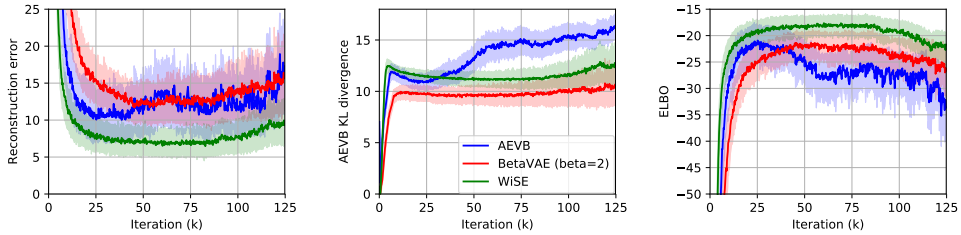


Figure 6: Comparison of training reconstruction error, AEVB KL divergence loss and ELBO given by AEVB, β-VAE and our WiSE-ALE methods on the sine wave dataset over 50 epochs (batch size = 64).

6 Conclusion and Future Work

In this paper, we derive a variational lower bound to the data log likelihood, which allows us to impose a prior constraint on the *bulk statistics* of the aggregate posterior distribution for the entire dataset. Using an analytic approximation to this lower bound as our learning objective, we propose WiSE-ALE algorithm. We have demonstrated its ability to achieve excellent reconstruction quality, as well as forming a smooth, compact and meaningful latent representation.

In the future, we are planning to analyse the error introduced in our approximation to our proposed lower bound. Further, we would like to investigate the potential to use the evaluation of the reconstruction likelihood term given by Eq. 15 as our learning objective, which will keep the lower bound property of our proposal and guarantee that our proposed posterior approaches the true posterior through optimisation.

References

Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828, August 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2013.50. URL `http://dx.doi.org/10.1109/TPAMI.2013.50`.

Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J. Davison. Codeslam learning a compact, optimisable representation for dense visual slam. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Ronald Clark, Michael Bloesch, Jan Czarnowski, Stefan Leutenegger, and Andrew J. Davison. Learning to solve nonlinear least squares for monocular stereo. In *The European Conference on Computer Vision (ECCV)*, September 2018.

Adam D. Cobb, Stephen J. Roberts, and Yarin Gal. Loss-calibrated approximate inference in bayesian neural networks. *CoRR*, abs/1805.03901, 2018.

Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016.

Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013. URL `http://arxiv.org/abs/1312.6114`.

Simon Lacoste-Julien, Ferenc Huszar, and Zoubin Ghahramani. Approximate inference for the loss-calibrated bayesian. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pp. 416–424, 2011. URL `http://www.jmlr.org/proceedings/papers/v15/lacoste_julien11a/lacoste_julien11a.pdf`.

Y. LeCun. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 1998. URL `https://ci.nii.ac.jp/naid/10027939599/en/`.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pp. 3730–3738, Washington, DC, USA, 2015. IEEE Computer Society. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.425. URL `http://dx.doi.org/10.1109/ICCV.2015.425`.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial autoencoders. In *International Conference on Learning Representations*, 2016. URL `http://arxiv.org/abs/1511.05644`.

Giorgio Patrini, Marcello Carioni, Patrick Forré, Samarth Bhargav, Max Welling, Rianne van den Berg, Tim Genewein, and Frank Nielsen. Sinkhorn autoencoders. *CoRR*, abs/1810.01118, 2018.

Ruslan Salakhutdinov and Geoffrey E. Hinton. Deep boltzmann machines. In *AISTATS*, volume 5 of *JMLR Proceedings*, pp. 448–455. JMLR.org, 2009.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. pp. 368–377, 1999.

Ilya O. Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schölkopf. Wasserstein auto-encoders. *CoRR*, abs/1711.01558, 2017.

C Villani. *Optimal transport – Old and new*, volume 338, pp. xxii+973. 01 2008. doi: 10.1007/978-3-540-71050-9.

APPENDIX FOR WISE

In this appendix, we omit the trainable parameters ϕ and θ in the expressions of distributions for simplicity. For example, $q(\mathbf{z}|\boldsymbol{x})$ is equivalent to $q_\phi(\mathbf{z}|\boldsymbol{x})$ and $p(\mathbf{x}|\boldsymbol{z})$ represents $p_\theta(\mathbf{x}|\boldsymbol{z})$.

A APPROXIMATION OF THE RECONSTRUCTION TERM

Here we demonstration that the reconstruction term $\mathbb{E}_{q(\mathbf{z}|\mathcal{D}_N)}\big[\log p(\mathcal{D}_N|\boldsymbol{z})\big]$ in our lower bound can be computed with individual sample likelihood $\log p(\boldsymbol{x}^{(i)}|\boldsymbol{z})$ and how our reconstruction error term becomes the same as the reconstruction term in the AEVB objective.

Firstly, we can substitute

$$q(\mathbf{z}|\mathcal{D}_N) = \frac{1}{N}\sum_{i=1}^{N} q(\mathbf{z}|\boldsymbol{x}^{(i)}) \tag{1}$$

into the reconstruction term $\mathbb{E}_{q(\mathbf{z}|\mathcal{D}_N)}\big[\log p(\mathcal{D}_N|\boldsymbol{z})\big]$, i.e.

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z}|\mathcal{D}_N)}\big[\log p(\mathcal{D}_N|\boldsymbol{z})\big] &= \int q(\mathbf{z}|\mathcal{D}_N)\big[\log p(\mathcal{D}_N|\boldsymbol{z})\big]\mathrm{d}\boldsymbol{z} \\
&= \int \frac{1}{N}\sum_{i=1}^{N} q(\mathbf{z}|\boldsymbol{x}^{(i)})\big[\log p(\mathcal{D}_N|\boldsymbol{z})\big]\mathrm{d}\boldsymbol{z} \\
&= \frac{1}{N}\sum_{i=1}^{N} \int q(\mathbf{z}|\boldsymbol{x}^{(i)})\big[\log p(\mathcal{D}_N|\boldsymbol{z})\big]\mathrm{d}\boldsymbol{z} \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathcal{D}_N|\boldsymbol{z})\big].
\end{aligned}$$

Now we can decompose the the marginal likelihood of the entire dataset as a product of individual samples, due to the conditional independence, i.e.

$$\log p(\mathcal{D}_N|\boldsymbol{z}) = \log \prod_{j=1}^{N} p(\boldsymbol{x}^{(j)}|\boldsymbol{z}) = \sum_{j=1}^{N} \log p(\boldsymbol{x}^{(j)}|\boldsymbol{z}).$$

Substituting this into the reconstruction term, we have:

$$\mathbb{E}_{q(\mathbf{z}|\mathcal{D}_N)}[\log p(\mathcal{D}_N|\boldsymbol{z})] = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\left[\sum_{j=1}^{N} \log p(\boldsymbol{x}^{(j)}|\boldsymbol{z})\right].$$

To evaluate the reconstruction term in our lower bound, we need to do the following: 1) draw a sample $\boldsymbol{x}^{(i)}$ from the dataset $\mathcal{D}_N$; 2) evaluate the latent code distribution $q(\mathbf{z}|\boldsymbol{x}^{(i)})$ through the encoder function $q(\cdot|\boldsymbol{x}^{(i)})$; 3) draw samples of $\mathbf{z}$ according to $q(\mathbf{z}|\boldsymbol{x}^{(i)})$; 4) reconstruct input samples using the sampled latent codes $\boldsymbol{z}^{(l)}$; 5) compute the reconstruction error w.r.t to every single input sample and sum this error.

We can simplify the above evaluation. Firstly, the sampling process in Step 3 can be replaced to a sampling process at the input using the reparameterisation trick. Besides, the sum of reconstruction errors w.r.t. all the input samples can be further simplified. To do this, we need to re-arrange the above expression as

$$\mathbb{E}_{q(\mathbf{z}|\mathcal{D}_N)}[\log p(\mathcal{D}_N|\boldsymbol{z})] = \sum_{i=1}^{N} \mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\left[\frac{1}{N}\sum_{j=1}^{N} \log p(\boldsymbol{x}^{(j)}|\boldsymbol{z})\right]$$

and apply Jensen inequality for the special case of $\log$, i.e.

$$\log\Big(\frac{1}{N}\sum_{i=1}^{N} a_i\Big) \geq \frac{1}{N}\sum_{i=1}^{N}\log\big(a_i\big)$$

to the terms inside the expectation. As a result, we have obtain an upper bound of the reconstruction error term as

$$\mathbb{E}_{q(\mathbf{z}|\mathcal{D}_N)}[\log p(\mathcal{D}_N|\boldsymbol{z})] \leq \sum_{i=1}^{N}\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\left[\log\left(\frac{1}{N}\sum_{j=1}^{N} p(\boldsymbol{x}^{(j)}|\boldsymbol{z})\right)\right].$$

This upper bound can be evaluated more efficiently with the assumption that the likelihood $p(\boldsymbol{x}^{(j)}|\boldsymbol{z})$ representing the probability of a reconstructed sample from a latent code $\boldsymbol{z}$ imitating the sample $\boldsymbol{x}^{(j)}$ will only be non-zero if $\boldsymbol{z}$ is sampled from the embedding prediction distribution with the same sample $\boldsymbol{x}^{(j)}$ at the encoder input. With this assumption, $N-1$ posterior distributions in the inner summation will be dropped as zeros and the only non-zero term is $p(\boldsymbol{x}^{(i)}|\boldsymbol{z})$. Therefore, the upper bound becomes

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z}|\mathcal{D}_N)}[\log p(\mathcal{D}_N|\boldsymbol{z})] &\leq \sum_{i=1}^{N}\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\left[\log\left(\frac{1}{N}p(\boldsymbol{x}^{(i)}|\boldsymbol{z})\right)\right] \\
&= \sum_{i=1}^{N}\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\Big[\log p(\boldsymbol{x}^{(i)}|\boldsymbol{z})\Big] - N\log N \\
&\approx \sum_{i=1}^{N}\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\Big[\log p(\boldsymbol{x}^{(i)}|\boldsymbol{z})\Big].
\end{aligned}$$

The constant can be omitted, because it will not affect the gradient updates of the parameters.

B An Upper Bound Approximation of the KL Term

$$\begin{aligned}
D_{\mathrm{KL}}\big[\, q(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big] &= \int q(\mathbf{z}|\mathcal{D}_N)\Big(\log q(\mathbf{z}|\mathcal{D}_N) - \log p(z)\Big)\mathrm{d}\boldsymbol{z} \\
&= \int \frac{1}{N}\sum_{i=1}^{N} q(\mathbf{z}|\boldsymbol{x}^{(i)})\Big(\log q(\mathbf{z}|\mathcal{D}_N) - \log p(\mathbf{z})\Big)\mathrm{d}\boldsymbol{z} \\
&= \frac{1}{N}\sum_{i=1}^{N}\int q(\mathbf{z}|\boldsymbol{x}^{(i)})\Big(\log q(\mathbf{z}|\mathcal{D}_N) - \log p(\mathbf{z})\Big)\mathrm{d}\boldsymbol{z} \\
&= \frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log q(\mathbf{z}|\mathcal{D}_N)\big] - \mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big]\Big)
\end{aligned}$$

Applying Jensen inequality, i.e.

$$\mathbb{E}_x\big[\log f(x)\big] \leq \log \mathbb{E}_x\big[f(x)\big], \tag{2}$$

to the first term of above equation, we have

$$D_{\mathrm{KL}}\big[\, q(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big] \;\leq\; \frac{1}{N}\sum_{i=1}^{N}\Big(\log \mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[q(\mathbf{z}|\mathcal{D}_N)\big]\Big) \;-\; \frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big]\Big).$$

We will look at the two summation individually. The expectation w.r.t. the aggregate posterior can be expanded as

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[q(\mathbf{z}|\mathcal{D}_N)\big] \;&=\; \int q(\mathbf{z}|\boldsymbol{x}^{(i)})\,\frac{1}{N}\sum_{j=1}^{N} q(\mathbf{z}|\boldsymbol{x}^{(j)})\,\mathrm{d}\boldsymbol{z} \\
&=\; \frac{1}{N}\sum_{j=1}^{N}\int q(\mathbf{z}|\boldsymbol{x}^{(i)})\,q(\mathbf{z}|\boldsymbol{x}^{(j)})\,\mathrm{d}\boldsymbol{z}.
\end{aligned}$$

We assume the posterior distribution of the latent code $\boldsymbol{z}$ given a specific input sample $\boldsymbol{x}^{(i)}$ is a diagonal Gaussian, i.e.

$$q(\mathbf{z} \,|\, \boldsymbol{x}^{(i)}) = \mathcal{N}\Big(\mathbf{z} \,|\, \mu^{(i)}, (\sigma^{(i)})^2\Big) = \prod_{k=1}^{d_z} \mathcal{N}\Big(\mathbf{z}_k \,|\, \mu_k^{(i)}, (\sigma_k^{(i)})^2\Big). \tag{3}$$

Similarly,

$$q(\mathbf{z} \,|\, \boldsymbol{x}^{(j)}) = \mathcal{N}\Big(\mathbf{z} \,|\, \mu^{(j)}, (\sigma^{(j)})^2\Big) = \prod_{k=1}^{d_z} \mathcal{N}\Big(\mathbf{z}_k \,|\, \mu_k^{(j)}, (\sigma_k^{(j)})^2\Big).$$

Therefore,

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[q(\mathbf{z}|\mathcal{D}_N)\big] &= \frac{1}{N}\sum_{j=1}^{N} \int \prod_{k=1}^{d_z} \mathcal{N}\Big(\mathbf{z}_k \,|\, \mu_k^{(i)}, (\sigma_k^{(i)})^2\Big) \prod_{k=1}^{d_z} \mathcal{N}\Big(\mathbf{z}_k \,|\, \mu_k^{(j)}, (\sigma_k^{(j)})^2\Big) \prod_{k=1}^{d_z} \mathrm{d}z_k \\
&= \frac{1}{N}\sum_{j=1}^{N} \prod_{k=1}^{d_z} \int \mathcal{N}\Big(\mathbf{z}_k \,|\, \mu_k^{(i)}, (\sigma_k^{(i)})^2\Big) \mathcal{N}\Big(\mathbf{z}_k \,|\, \mu_k^{(j)}, (\sigma_k^{(j)})^2\Big) \mathrm{d}\boldsymbol{z}_k.
\end{aligned}$$

Substituting the exponential form for Gaussian distribution, i.e.

$$\mathcal{N}\Big(\mathbf{z}_k \,|\, \mu_k^{(i)}, (\sigma_k^{(i)})^2\Big) = \frac{1}{\sqrt{2\pi(\sigma_k^{(i)})^2}} \exp\left(-\frac{(z_k - \mu_k^{(i)})^2}{2(\sigma_k^{(i)})^2}\right), \tag{4}$$

to the above equation, we have

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[q(\mathbf{z}|\mathcal{D}_N)\big] &= \frac{1}{N}\sum_{j=1}^{N} \prod_{k=1}^{d_z} \int \frac{1}{2\pi\sigma_k^{(i)}\sigma_k^{(j)}} \exp\left(-\frac{(z_k - \mu_k^{(i)})^2}{2(\sigma_k^{(i)})^2} - \frac{(z_k - \mu_k^{(j)})^2}{2(\sigma_k^{(j)})^2}\right) \mathrm{d}z_k \\
&= \frac{1}{N}\sum_{j=1}^{N} \prod_{k=1}^{d_z} \frac{1}{2\pi\sigma_k^{(i)}\sigma_k^{(j)}} \int \exp\left(-\frac{(z_k - \mu_k^{(i)})^2}{2(\sigma_k^{(i)})^2} - \frac{(z_k - \mu_k^{(j)})^2}{2(\sigma_k^{(j)})^2}\right) \mathrm{d}z_k.
\end{aligned}$$

The exponent of the above equation can be simplified to

$$\begin{aligned}
&-\frac{(z_k - \mu_k^{(i)})^2}{2(\sigma_k^{(i)})^2} - \frac{(z_k - \mu_k^{(j)})^2}{2(\sigma_k^{(j)})^2} \\
&= -\frac{1}{2}\left(\frac{1}{(\sigma_k^{(i)})^2} + \frac{1}{(\sigma_k^{(j)})^2}\right) z_k^2 + \left(\frac{\mu_k^{(i)}}{(\sigma_k^{(i)})^2} + \frac{\mu_k^{(j)}}{(\sigma_k^{(j)})^2}\right) z_k - \frac{1}{2}\left(\frac{(\mu_k^{(i)})^2}{(\sigma_k^{(i)})^2} + \frac{(\mu_k^{(i)})^2}{(\sigma_k^{(i)})^2}\right).
\end{aligned}$$

Using the following properties, i.e.

$$\int_{-\infty}^{\infty} \exp\big(-ax^2 + bx\big)\,\mathrm{d}x = \sqrt{\frac{\pi}{a}} \exp\left(\frac{b^2}{4a}\right) \quad (a \geq 0), \tag{5}$$

we can evaluate the integral needed for $\mathbb{E}_{q(z|x^{(i)})}\big[q(z|\mathcal{D}_N)\big]$ as

$$\begin{aligned}
&\frac{1}{2\pi\sigma_k^{(i)}\sigma_k^{(j)}} \int_{z_k} \exp\left(-\frac{(z_k - \mu_k^{(i)})^2}{2(\sigma_k^{(i)})^2} - \frac{(z_k - \mu_k^{(j)})^2}{2(\sigma_k^{(j)})^2}\right) \mathrm{d}z_k \\
&\qquad = \frac{1}{\sqrt{2\pi\big((\sigma_k^{(i)})^2 + (\sigma_k^{(j)})^2\big)}} \exp\left(-\frac{1}{2}\frac{(\mu_k^{(i)} - \mu_k^{(j)})^2}{(\sigma_k^{(i)})^2 + (\sigma_k^{(j)})^2}\right).
\end{aligned}$$

Therefore, we have obtained the expression for the first term in our upper bound, i.e.

$$\frac{1}{N}\sum_{i=1}^{N}\Big(\log \mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[q(\mathbf{z}|\mathcal{D}_N)\big]\Big) \tag{6}$$

$$= \frac{1}{N}\sum_{i=1}^{N} \log\left(\frac{1}{N}\sum_{j=1}^{N}\prod_{k=1}^{d_z} \frac{1}{\sqrt{2\pi\big((\sigma_k^{(i)})^2 + (\sigma_k^{(j)})^2\big)}} \exp\left(-\frac{1}{2}\frac{(\mu_k^{(i)} - \mu_k^{(j)})^2}{(\sigma_k^{(i)})^2 + (\sigma_k^{(j)})^2}\right)\right). \tag{7}$$

To find out the expression for the second term $\frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big]\Big)$, we first examine the prior distribution $p(\mathbf{z})$ which is chosen to be a zero-mean unit-variance Gaussian across all latent code dimensions, i.e.

$$p(\mathbf{z}) \;=\; \mathcal{N}\big(\mathbf{z}\,|\,0,\;\mathcal{I}\big) \;=\; \prod_{k=1}^{d_z}\mathcal{N}\big(\mathbf{z}_k\,|\,0,1\big). \tag{8}$$

Therefore,

$$\log p(\mathbf{z}) = \sum_{k=1}^{d_z}\log\mathcal{N}\big(\mathbf{z}_k\,|\,0,1\big) = -\frac{1}{2}\sum_{k=1}^{d_z}\Big(\log\big(2\pi\big)+z_k^2\Big) \tag{9}$$

Substituting this expression for $\log p(\mathbf{z})$ into $\frac{1}{N}\sum_{i=1}^{N}\Big(\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big]\Big)$ and examining the expectation term for now, we have

$$\begin{aligned}
\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big] \;&=\; \sum_{k=1}^{d_z}\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z}_k)\big]\\
&=\; \sum_{k=1}^{d_z}\mathbb{E}_{q(\mathbf{z}_k|\boldsymbol{x}^{(i)})q(\mathbf{z}_{\setminus k}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z}_k)\big]\\
&=\; \sum_{k=1}^{d_z}\mathbb{E}_{q(\mathbf{z}_k|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z}_k)\big]\\
&=\; \sum_{k=1}^{d_z}\int q(\mathbf{z}_k|\boldsymbol{x}^{(i)})\log p(\mathbf{z}_k)\,\mathrm{d}z_k\\
&=\; -\frac{1}{2}\sum_{k=1}^{d_z}\int q(\mathbf{z}_k|\boldsymbol{x}^{(i)})\Big(\log\big(2\pi\big)+z_k^2\Big)\,\mathrm{d}z_k\\
&=\; -\frac{1}{2}\sum_{k=1}^{d_z}\Bigg(\log\big(2\pi\big)\int q(\mathbf{z}_k|\boldsymbol{x}^{(i)})\,\mathrm{d}z_k \;+\; \int q(\mathbf{z}_k|\boldsymbol{x}^{(i)})z_k^2\,\mathrm{d}z_k\Bigg).
\end{aligned}$$

The first integral $\int q(\mathbf{z}_k|\boldsymbol{x}^{(i)})\,\mathrm{d}z_k = 1$. To evaluate the second integral, we substitute Equation (4) and use the following properties, i.e.

$$\int_{-\infty}^{\infty}\exp\big(-ax^2\big)\,\mathrm{d}x \;=\; \frac{1}{2}\sqrt{\frac{\pi}{a}},\quad a\geq 0 \tag{10}$$

$$\int_{-\infty}^{\infty}x\exp\big(-a(x-b)^2\big)\,\mathrm{d}x \;=\; b\sqrt{\frac{\pi}{a}},\quad \mathrm{Re}(a)\geq 0 \tag{11}$$

$$\int_{-\infty}^{\infty}x^2\exp\big(-ax^2\big)\,\mathrm{d}x \;=\; \frac{1}{2}\sqrt{\frac{\pi}{a^3}},\quad a\geq 0. \tag{12}$$

As a result, we have

$$\int_{z_k}q(z_k|\boldsymbol{x}^{(i)})z_k^2\,\mathrm{d}z_k \;=\; \big(\sigma_k^{(i)}\big)^2+\big(\mu_k^{(i)}\big)^2.$$

Therefore,

$$\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big] \;=\; -\frac{1}{2}\sum_{k=1}^{d_z}\Bigg(\big(\sigma_k^{(i)}\big)^2+\big(\mu_k^{(i)}\big)^2+\log 2\pi\Bigg) \tag{13}$$

$$\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}_{q(\mathbf{z}|\boldsymbol{x}^{(i)})}\big[\log p(\mathbf{z})\big] \;=\; -\frac{1}{2N}\sum_{i=1}^{N}\sum_{k=1}^{d_z}\Bigg(\big(\sigma_k^{(i)}\big)^2+\big(\mu_k^{(i)}\big)^2+\log 2\pi\Bigg). \tag{14}$$

Combining the first term defined in Equation (6) and the second term defined in Equation (13), we have obtained the expression for the overall upper bound as

$$
\begin{aligned}
&D_{\mathrm{KL}}\big[\, q(\mathbf{z}|\mathcal{D}_N)\|p(\mathbf{z})\big] \\
&\quad\leq\ \frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{N}\sum_{j=1}^{N}\prod_{k=1}^{d_z}\frac{1}{\sqrt{2\pi\big(\big(\sigma_k^{(i)}\big)^2+\big(\sigma_k^{(j)}\big)^2\big)}}\exp\left(-\frac{1}{2}\frac{\big(\mu_k^{(i)}-\mu_k^{(j)}\big)^2}{\big(\sigma_k^{(i)}\big)^2+\big(\sigma_k^{(j)}\big)^2}\right)\right) \\
&\quad+\ \frac{1}{2N}\sum_{i=1}^{N}\sum_{k=1}^{d_z}\left(\big(\sigma_k^{(i)}\big)^2+\big(\mu_k^{(i)}\big)^2+\log 2\pi\right).
\end{aligned}
$$

C WiSE Algorithm

Algorithm 1 WiSE algorithm. Either $L_{\text{approx}}^{\text{WiSE}}(\phi,\theta;\mathcal{B}_M)$ defined in Eq. 19 in Section 3.5 can be used as the learning objective function.

ϕ, $\theta \leftarrow$ Initialize parameters
repeat
  $\mathcal{B}_M \leftarrow$ Draw M random samples from $\mathcal{D}_N$
  $\epsilon \leftarrow \mathcal{N}(0,\boldsymbol{I})$ Apply reparameterisation trick so that $\boldsymbol{z}\sim q_\phi(\mathbf{z}|\boldsymbol{x}^{(i)})$ becomes $\boldsymbol{z}=\mu^{(i)}+\epsilon\odot\sigma^{(i)}$

  $g \leftarrow \bigtriangledown_{\phi,\theta} L_{\text{approx}}^{\text{WiSE}}(\phi,\theta;\mathcal{B}_M)$ Compute the gradient
  ϕ, $\theta \leftarrow$ Update parameters using g according to AdamOptimizer
until convergence of the objective function or end of iterations
return ϕ,θ

D Experiment Details

We carry out experiments on four datasets (Sine wave, MNIST, Teapot and CelebA) to examine different properties of the latent representation learnt from the proposed WiSE algorithm. Specifically, we compare with β-VAE on the smoothness and disentanglement of the learnt representation and compare with WAE and AEVB on the reconstruction quality. In addition, by learning a 2D embedding of the MNIST dataset, we are able to visualise the latent embedding distributions learnt from AEVB, β-VAE, WAE and our WiSE and compare the compactness and smoothness of the learnt latent space across these methods. Here we give the implementation details for each dataset.

D.1 Sine Wave

We aim to learn a latent representation in $\mathcal{R}^4$ for a one second long sine wave with sampling rate of 256Hz. The network architecture for the Sine wave dataset is shown below. $\boldsymbol{x}$ is an input sample, μ and σ are the latent code mean and latent code standard deviation to define the embedding distribution $q(\mathbf{z}|\boldsymbol{x})$ and $\hat{\boldsymbol{x}}$ is the reconstructed input sample. ϵ is an auxiliary variable drawn from unit Gaussian at the input of the encoder network so that an estimate of a sample from the embedding distribution $q(\mathbf{z}|\boldsymbol{x})$ can be computed. $\text{Conv}_k^{m\times n}$ denotes a convolution operation with k filters each of size $m\times n$. $\text{TransposedConv}_k^{m\times n}$ (stride = (a,b)) denotes a stride of a and b for the sliding window and k filters each of size $m\times n$. FC_k denotes a fully connected layer with output in $\mathcal{R}^k$. Reshape_a^b denotes reshaping an variable from dimension a to dimension b. ReLU denotes rectified linear units.

Encoder network:

$$
\begin{aligned}
\boldsymbol{x} \in \mathcal{R}^{256\times 1} &\rightarrow \text{Conv}_{16}^{16\times 1}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}_{16}^{16\times 1}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}_{32}^{16\times 1}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}_{32}^{16\times 1}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}_{64}^{8\times 1}\ (\text{stride}=2) \rightarrow \text{ReLU} \rightarrow \text{FC}_{64} \rightarrow \text{FC}_4 \Rightarrow\ \mu \in \mathcal{R}^4 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \searrow \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{FC}_4 \rightarrow \text{ReLU} \Rightarrow\ \sigma \in \mathcal{R}^4
\end{aligned}
$$

Decoder network:

$$
\begin{aligned}
\boldsymbol{z} = \mu + \epsilon \odot \sigma &\rightarrow \text{FC}_{16} \rightarrow \text{ReLU} \rightarrow \text{Reshape}_{16}^{1\times 1\times 16} \\
&\rightarrow \text{Conv}_{128}^{1\times 1} \rightarrow \text{ReLU} \\
&\rightarrow \text{TransposedConv}_{64}^{8\times 1}\ (\text{stride}=(4,1)) \rightarrow \text{ReLU} \\
&\rightarrow \text{TransposedConv}_{32}^{16\times 1}\ (\text{stride}=(4,1)) \rightarrow \text{ReLU} \\
&\rightarrow \text{TransposedConv}_{16}^{16\times 1}\ (\text{stride}=(4,1)) \rightarrow \text{ReLU} \\
&\rightarrow \text{TransposedConv}_{1}^{16\times 1}\ (\text{stride}=(4,1)) \rightarrow \text{ReLU} \rightarrow \text{Reshape}_{256\times 1\times 1}^{256\times 1} \Rightarrow\ \hat{\boldsymbol{x}} \in \mathcal{R}^{256\times 1}
\end{aligned}
$$

We use the following hyper-parameters to train the network:

Batch size	**Number of epochs**	**Optimizer**	**Learning rate**	**Padding**
64	50	Adam	5×10^{-4}	SAME

D.2 MNIST

We aim to learn a 2D embedding of the MNIST dataset. The network architecture is shown below.

Encoder network:

$$
\begin{aligned}
\boldsymbol{x} \in \mathcal{R}^{28\times 28\times 1} &\rightarrow \text{Conv}_{16}^{4\times 4}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}_{32}^{4\times 4}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}_{64}^{4\times 4}\ (\text{stride}=2) \rightarrow \text{ReLU} \rightarrow \text{FC}_{32} \rightarrow \text{FC}_2 \Rightarrow\ \mu \in \mathcal{R}^2 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \searrow \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{FC}_2 \rightarrow \text{ReLU} \Rightarrow\ \sigma \in \mathcal{R}^2
\end{aligned}
$$

Decoder network:

$$
\begin{aligned}
\boldsymbol{z} = \mu + \epsilon \odot \sigma &\rightarrow \text{FC}_{16} \rightarrow \text{ReLU} \\
&\rightarrow \text{FC}_{128} \rightarrow \text{ReLU} \\
&\rightarrow \text{FC}_{784} \rightarrow \text{Sigmoid} \rightarrow \text{Reshape}_{784}^{28\times 28\times 1} \Rightarrow\ \hat{\boldsymbol{x}} \in \mathcal{R}^{28\times 28\times 1}
\end{aligned}
$$

We use the following hyper-parameters to train the network:

Batch size	**Number of epochs**	**Optimizer**	**Learning rate**	**Padding**
64	30	Adam	1×10^{-3}	SAME

D.3 CELEBA

We implement our WiSE and AEVB on the same encoder and decoder network used in WAE in order to compare the reconstruction quality of our method with AEVB and WAE. The network architecture and training parameters are stated below.

Encoder network:

$$
\begin{aligned}
\boldsymbol{x} \in \mathcal{R}^{64\times64\times3} &\rightarrow \text{Conv}^{5\times5}_{128}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}^{5\times5}_{256}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}^{5\times5}_{512}\ (\text{stride}=2) \rightarrow \text{ReLU} \\
&\rightarrow \text{Conv}^{5\times5}_{1024}\ (\text{stride}=2) \rightarrow \text{ReLU} \rightarrow \text{FC}_{64} \Rightarrow\ \mu \in \mathcal{R}^{64} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \searrow \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{FC}_{64} \Rightarrow\ \sigma \in \mathcal{R}^{64}
\end{aligned}
$$

Decoder network:

$$
\begin{aligned}
\boldsymbol{z} = \mu + \epsilon \odot \sigma &\rightarrow \text{FC}_{64\times1024} \rightarrow \text{ReLU} \rightarrow \text{Reshape}^{8\times8\times1024}_{64\times1024} \\
&\rightarrow \text{TransposedConv}^{5\times5}_{512}\ (\text{stride}=(2,2)) \rightarrow \text{BN} \rightarrow \text{ReLU} \\
&\rightarrow \text{TransposedConv}^{5\times5}_{256}\ (\text{stride}=(2,2)) \rightarrow \text{BN} \rightarrow \text{ReLU} \\
&\rightarrow \text{TransposedConv}^{5\times5}_{128}\ (\text{stride}=(2,2)) \rightarrow \text{BN} \rightarrow \text{ReLU} \\
&\rightarrow \text{TransposedConv}^{5\times5}_{1}\ (\text{stride}=(1,1)) \Rightarrow\ \hat{\boldsymbol{x}} \in \mathcal{R}^{64\times64\times3}
\end{aligned}
$$

We use the following hyper-parameters to train the network:

Batch size	**Number of epochs**	**Optimizer**	**Learning rate**	**Padding**
256	50	Adam	3×10^{-4} at the start, 1.5×10^{-4} after 30 epochs	SAME