AE + VAE - Rough Overview

- The principal aim of an autoencoder is to find a meaningful low-dimensional (relatively speaking) representation of data
- e.g. PCA could be viewed as a naive autoencoder, capturing linear features.

- Given a density γ, denote by γ^N the empirical density,

$$\gamma^N(u) := \frac{1}{N} \sum_{i=1}^{N} \delta(u - u^{(i)})$$

↑ sampled point.

- The aim is to find an approximate factorisation of Id which is accurate on the support of γ, given access only to γ^N.

Let $f : \mathbb{R}^d \times \Theta_f \to \mathbb{R}^{d_z}$
$g : \mathbb{R}^{d_z} \times \Theta_g \to \mathbb{R}^d$.

e.g., f and g can be NNs with parameters living in the spaces $\Theta_f \subseteq \mathbb{R}^{p_f}$ and $\Theta_g \subseteq \mathbb{R}^{p_g}$, respectively

- Then simply define the reconstruction loss

$$J(\theta_f, \theta_g) = \| Id - g(f(\cdot; \theta_f)\theta_g) \|^2_{H_\gamma}$$

$$:= \mathbb{E}_{u \sim \gamma}\left[|u - g(f(u; \theta_f)\theta_g)|^2 \right]$$

Obviously we have to compute this expectation w.r.t. γ^N in reality.

We thus have the objective

$$(\theta_f^*, \theta_g^*) \in \underset{(\theta_f, \theta_g)}{\arg\min} \; J(\theta_f, \theta_g),$$

and write $f^*(u) = f(u; \theta_f^*)$ and $g^*(u) = g(u; \theta_g^*)$

- Note that we have converted an unsupervised learning problem into a supervised learning problem.
- Some benefits include robustness, computational tractability, and interpretability.

PCA Example

We elaborate a bit on the 'PCA as an autoencoder' comment.

Let $m := \mathbb{E}[u]$ under $u \sim Y$

and $C := \mathbb{E}[(u-m) \otimes (u-m)]$ under $u \sim Y$.

($\otimes$ denotes the Kronecker / tensor product, so $(C)_{ij} = \mathbb{E}[(u_i - m_i)(u_j - m_j)] = (C)_{ji}$)

PCA consists of finding at its first step

$$w_1 = \underset{\|w\|=1}{\arg\max} \left\{ \sum_i \langle u^i, w \rangle^2 \right\}$$

$$= \arg\max \left\{ \frac{w^T U^T U w}{w^T w} \right\},$$

where U is the mean-shifted data matrix.
i.e. $U^T U = C$, the covariance matrix.

This is maximised when w is the eigenvector corresponding to the largest eigenvalue. The point is to transfer to a coordinate system where the greatest variance of the data among all subspace projections is given by that spanned by $\{w_i\}_{i=1}^k$ for k the desired # of components.

To get further components are subtract the first component from the data matrix, $U_2 = U - U w_1 w_1^T$, and repeat the process.

So, let $(w^i, \lambda^i) \in \mathbb{R}^d \times \mathbb{R}^+$ be the eigenpairs (normalised) s.t. $\lambda_1 > \lambda_2 > \cdots > \lambda_n$. Note that C is symmetric and PSD, so all eigenvalues are positive and there exists an orthonormal basis of eigenvectors.

For $d_2 \leq d$, define

$$f(u) = (\langle u, w^1 \rangle, \cdots, \langle u, w_{d_2} \rangle) \in \mathbb{R}^{d_2}$$

$$g(z) = \sum_{j=1}^{d_2} z_j w_j$$

$$\longrightarrow g(f(u)) = \left(\sum_{j=1}^{d_2} w_j \otimes w_j \right) u$$

i.e., this is a truncated version of the representation of Id given by the spectral theorem for PD symmetric matrices.

VAEs

- Define the pushforward map by

$$f_{\#}\mu(B) := \mu(f^{-1}(B)), \quad B \in \Sigma_2,$$

where $(X_1, \Sigma_1), (X_2, \Sigma_2)$ are measurable spaces, and f is μ-measurable map from X_1 to X_2, and $\mu : \Sigma_1 \to [0, +\infty]$

- Then

$$\hat{\gamma} = f^k_{\#} \gamma$$

gives the approximate distribution in the latent space, but

we can't specify ξ.

- We might want to impose some specified distribution in the latent space. This is the point of a VAE.

Definition :

* Given densities ρ and ξ on $\mathbb{R}^d$, a coupling is a density π on $\mathbb{R}^d \times \mathbb{R}^d$ s.t.

$$\int_{\mathbb{R}^d} \pi(z,u)\,du = \rho(z)$$

and

$$\int_{\mathbb{R}^d} \pi(z,u)\,dz = \xi(u)$$

So marginals of deliver ρ and ξ.

- Note that couplings are by no means unique.
- The extension to ρ and ξ measures over different dimensions is obvious.

- Let π be a coupling of γ and ξ. Then by definition, we have

$$\pi(u,z) = \underbrace{\mathbb{P}(u|z)}\,\xi(z) = \underbrace{\mathbb{P}(z|u)}\,\gamma(u)$$

→ The point of a VAE is to approximate the conditional densities by parametrised families.

- for example, we could have

$$u|z \sim \mathcal{N}(g(z;\theta_g), \sigma_g^2 \mathrm{Id})$$

$$z|u \sim \mathcal{N}(f(u;\theta_f), \sigma_f^2 \mathrm{Id})$$

the approximate coupling density becomes

two different approximations of the same thing!

$$\pi_g(u,z) = \frac{1}{Z_g} \exp\left(\frac{-1}{2\sigma_g^2}|u - g(z;\theta_g)|^2\right) \xi(z)$$

$$\pi_f(u,z) = \frac{1}{Z_f} \exp\left(\frac{-1}{2\sigma_f^2}|z - f(u;\theta_f)|^2\right) \gamma(u)$$

for simplicity, let's assume σ_f and σ_g are fixed and determine the parameters of f and g, meaning find θ_f and θ_g s.t. π_g and π_f are closed.

Definition:

* the KL-divergence of a distribution P from a distribution Q is

$$D_{KL}(P \| Q) = \int p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$$

Lemma:

* If $\rho = \mathcal{N}(\mu_\rho, \Sigma_\rho)$ and $\ell = \mathcal{N}(\mu_\ell, \Sigma_\ell)$, then

$$D_{KL}(\rho \| \ell) = \frac{1}{2}\left[\mathrm{Tr}(\Sigma_\ell^{-1}\Sigma_\rho) - d + (\mu_\rho - \mu_\ell)\Sigma_\ell^{-1}(\mu_\rho - \mu_\ell) + \log\left(\frac{\det \Sigma_\ell}{\det \Sigma_\rho}\right)\right]$$

One can check that $\gamma = \mathcal{N}(f(u;\theta_f), \sigma_f^2 \mathrm{Id})$

$$D_{KL}(\pi_f \| \pi_g) = \mathbb{E}^{u\sim\gamma}\left[\mathbb{E}^{z\sim P(z|u)}\left[\frac{1}{2\sigma_g^2}|u - g(z;\theta_g)|^2\right] + D_{KL}(P(\cdot|u) \| \xi)\right] + \text{const}$$

$$= \frac{1}{2}|f(u;\theta_f)|^2 \quad \text{by}$$

$$= \mathbb{E}^{u\sim r}\left[\mathbb{E}^{\xi\sim N(0,1)}\left[\frac{1}{2\sigma_g^2}\left|u - g\left(f(u;\theta_f) + \sigma_f \xi;\theta_g\right)\right|^2\right] + \frac{1}{2}\left|f(u;\theta_f)\right|^2\right]$$

This is a regularised version of the standard autoencoder.

VAEs in Greater Detail

- Assume that $x \sim p^*$, where we will just write the density as $p^*(x)$. This is a true distribution we need to model.
- We want a model class which is flexible enough to adapt to data whilst respecting a priori constraints. We denote our model by $p_\theta(x)$

- Denote DNNs as a vector function $\text{NeuralNet}(\cdot)$. We will be interested in using DNNs to approximate density functions.

Latent Variables

- Latent variables, z, are variables which we do not see, so we cannot infer their values directly.
- Our data model then represents a joint distribution, $p_\theta(x, z)$, over both observed and latent variables

- The marginal distribution is just

$$p_\theta(x) = \int p_\theta(x, z) dz,$$

also called the (single datapoint) marginal likelihood / model evidence.

- Deep latent variable Models are latent variable models for which $p_\theta(x, z)$ has distributions parametrised by NNs.
- The benefit is that we can have a simple directed model, like a parametrised Gaussian conditional, but build them up to be very expressive.
- Formally, this mean that

$$p_\theta(x, z) = p_\theta(z)\, p_\theta(x|z),$$

where one or both of the densities on the RHS is specified. $p(z)$ is the prior.

- The difficulty in training these models is that for DLVMs the marginal probability of data under the model is hard to estimate.
- This mean we can't differentiate w.r.t. θ and then optimize.
- VAEs are designed to address this problem.

Variational Autoencoders

- Note that $p_\theta(x,z)$ is tractable, and that

$$p_\theta(z|x) = \frac{p_\theta(x,z)}{p_\theta(x)}$$

So to get $p_\theta(x)$, can we find a good way of determining roughly what $p_\theta(z|x)$ is?

- Yes, introduce a parametric inference model, $q_\phi(z|x)$. This is called the encoder.
- Similarly to a DLVM, we can parametrize $q_\phi(z|x)$ by a NN. The variational parameters ϕ are the weights and biases of the NN.

e.g. $(\mu, \log(\sigma)) = \text{EncoderNeuralNet}_\phi(x)$

$$q_\phi(z|x) = \mathcal{N}(z; \mu, \sigma)$$

- So we see that for each point x we're trying to learn the parameters which give a variational distribution in the latent space.

The Evidence Lower Bound

- Most variational Bayesian methods try to maximise a quantity called the ELBO
- for any choice of inference model $q_\phi(z|x)$, we have

$$\log(p_\theta(x)) = \mathbb{E}_{z\sim q_\phi(z|x)}[\log(p_\theta(x))]$$

$$= \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{p_\theta(z|x)}\right)\right]$$

$$= \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)} \cdot \frac{q_\phi(z|x)}{p_\theta(z|x)}\right)\right]$$

$$= \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right)\right]$$

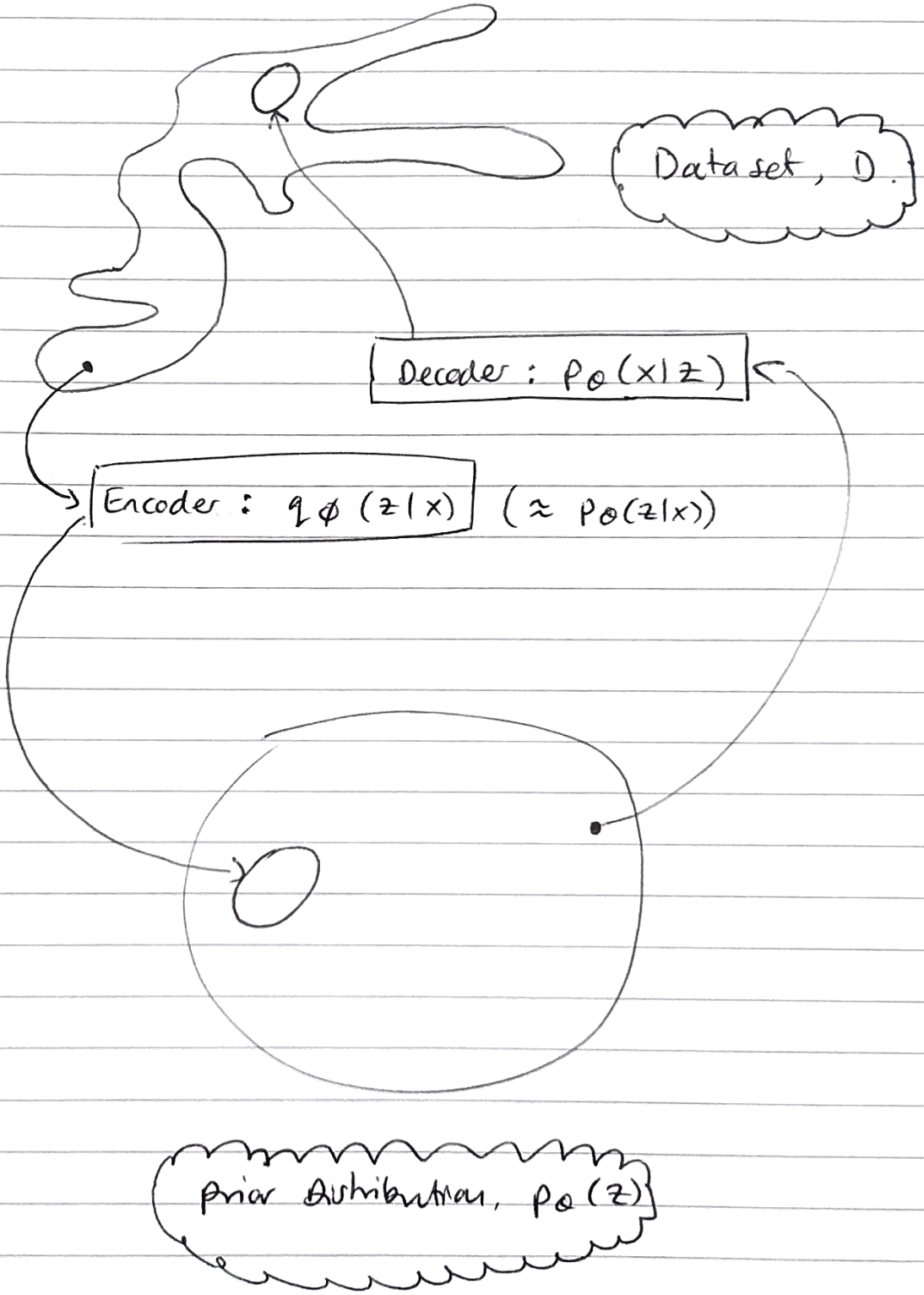
$$+ \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log\left(\frac{q_\phi(z|x)}{p_\theta(z|x)}\right)\right]$$

$$:= \underbrace{\mathcal{L}_{\theta,\phi}(x)}_{\text{ELBO}(x)} + \underbrace{D_{KL}\left(q_\phi(z|x) \,\|\, p_\theta(z|x)\right)}_{\geq 0},$$

so this is indeed a lower bound on the probability of that data x under the model p_θ.

- Note that we have equality iff $q_\phi(z|x)$ equals the true posterior distribution.
- So in addition to, by definition, quantifying the divergence of the approximate posterior from the true posterior, we can interpret D_{KL} as the gap between the ELBO and the marginal likelihood $p_\theta(x)$.

To summarise schematically, we are doing the following:



So we see that the generative model learns a joint distribution $p_\theta(x,z)$ factorised as $p_\theta(z)p_\theta(x|z)$, where we impose a prior $p_\theta(z)$ on the latent space.
The stochastic encoder $q_\phi(z|x)$ approximates the true but intractable posterior $p_\theta(z|x)$ of the model.

- we just use the ELBO, $\mathbb{E}[\log(p_\theta(x,z)) - \log(q_\phi(z|x))]$ under $z \sim q_\phi(z|x)$, as the optimisation objective, jointly finding the best parameters θ and ϕ, e.g. using SGD.
- As our previous note on the interpretation of the ELBO said, this optimisation gives us:
  a) Approximate maximisation of the marginal likelihood, $p_\theta(x)$, so our generative model improves
  b) Minimisation of the KL divergence of $q_\phi(z|x)$ from the true posterior, $p_\theta(z|x)$, making $q_\phi(z|x)$ better.

Estimating the Gradient

- The individual datapoint ELBO, as well as its gradient, $\nabla_{\theta,\phi} L_{\theta,\phi}(x)$, are normally intractable, but good estimators exist.
- For gradients w.r.t. the parameters of the generative model, write

$$\nabla_\theta L_{\theta,\phi}(x) = \nabla_\theta \mathbb{E}_{z\sim q_\phi(z|x)}[\log(p_\theta(x,z) - \log(q_\phi(z|x))]$$

$$= \mathbb{E}_{z\sim q_\phi(z|x)}[\nabla_\theta(\log(p_\theta(x,z)) - \log(q_\phi(z|x)))]$$

$$\simeq \nabla_\theta \log(p_\theta(x,z)),$$

where we use $\simeq$ to denote that the z at the end is a random sample of z. i.e., use Monte Carlo to estimate gradients.

- Gradients w.r.t. ϕ are more challenging, since the expectation is taken w.r.t. $q_\phi(z|x)$ which depends on ϕ. i.e., $\nabla_\phi \mathbb{E}_{z\sim q_\phi(z|x)}[\log(p_\theta(x,z)) - \log(q_\phi(z|x))]$ is not generally

the same as $\mathbb{E}_{z\sim q_\phi(z|x)}[-\nabla_\phi \log(q_\phi(z|x))]$.

- To deal with this problem, we introduce a new parametrization

Parametrization Trick

- Write $z \sim q_\phi(z|x)$ as the transformation of some other RV ϵ, given x and ϕ:

$$z = g(\epsilon, \phi, x),$$

where the distribution of ϵ is independent of x and ϕ.
- Note that g must be invertible and differentiable.

- Now we can write

$$\nabla_\phi \mathbb{E}_{z\sim q_\phi(z|x)}[f(z)] = \nabla_\phi \mathbb{E}_{\epsilon\sim p(\epsilon)}[f(z)]$$

$$= \mathbb{E}_{\epsilon\sim p(\epsilon)}[\nabla_\phi f(z)]$$

$$\simeq \nabla_\phi f(z),$$

for $z = g(\phi, x, \epsilon)$ and $\epsilon \sim p(\epsilon)$ a random sample.

- One can check that this single sample algorithm, in which we:

a) draw $\epsilon \sim p(\epsilon)$
b) set $z = g(\phi, x, \epsilon)$
c) estimate $\tilde{\mathcal{L}}_{\theta,\phi}(x) = \log(p_\theta(x,z)) - \log(q_\phi(z|x))$

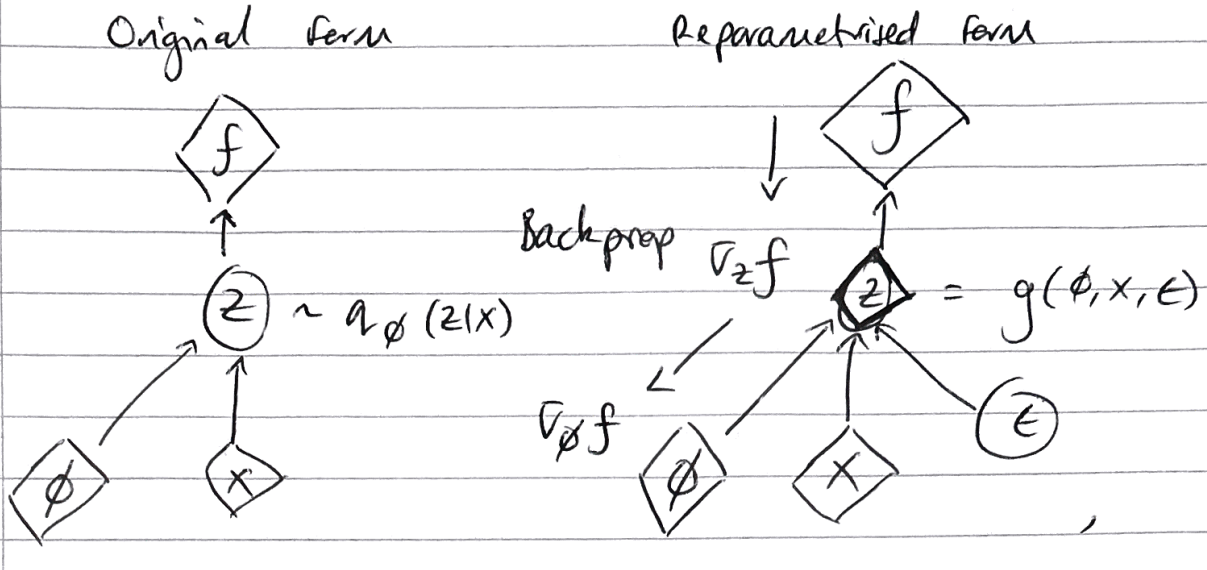
is an unbiased estimator of the gradient:

$$\mathbb{E}_{p(\epsilon)}[\nabla_{\theta,\phi} \tilde{\mathcal{L}}_{\theta,\phi}(x,\epsilon)] = \mathbb{E}_{z\sim p(\epsilon)}[\nabla_{\theta,\phi} \log(p_\theta(x,z)) - \nabla_{\theta,\phi} \log(q_\phi(z|x))]$$

$$= \nabla_{\theta,\phi} \mathbb{E}_{z\sim p(\epsilon)}[\log(p_\theta(x,z)) - \log(q_\phi(z|x))]$$

$$= \nabla_{\theta,\phi} \mathcal{L}_{\theta,\phi}(x).$$

Schematically, we have the following correction:



where ◇ = deterministic and ○ = random.

Computation of the ELBO

- To evaluate the estimator of the ELBO, we need the density, $\log(q_\phi(z|x))$. This is easy if we choose the right $q(\cdot)$, noting that typically are known (chosen, indeed) $p(\epsilon)$.
- If g is invertible, the densities of ϵ and z are related via

$$\log(q_\phi(z|x)) = \log(p(\epsilon)) - \log(|D_\phi(x,\epsilon)|),$$

for $D_\phi(x,\epsilon) = \frac{\partial z}{\partial \epsilon} = \begin{pmatrix} \frac{\partial z_1}{\partial \epsilon_1} & \cdots & \frac{\partial z_1}{\partial \epsilon_k} \\ \vdots & \ddots & \vdots \\ \frac{\partial z_k}{\partial \epsilon_1} & \cdots & \frac{\partial z_k}{\partial \epsilon_k} \end{pmatrix}$,

the Jacobian of the transformation $\epsilon \longmapsto z(\epsilon)$.

- The hope is that we can build transformations, $g(\cdot)$, which are flexible and for which $\log(|D_\phi(x,\epsilon)|)$ is easy to compute (hence giving a tractable and flexible

inference model, $q_\phi(z|x)$.)

An Example

- we will treat the case of a Gaussian with diagonal covariance matrix, and then the full case.

(1) we will say that

$$q_\phi(z|x) = \mathcal{N}(z; \mu, \mathrm{diag}(\sigma^2)),$$

where μ and σ^2 are functions of x:

$$(\mu, \log(\sigma)) = \text{Encoder Neural Net } (x)$$

$$\rightarrow q_\phi(z|x) = \prod_i q_\phi(z_i|x) = \prod_i \mathcal{N}(z_i; \mu_i, \sigma_i^2)$$

for the reparametrisation, write

$$\epsilon \sim \mathcal{N}(0, Id)$$

$$z = \mu + \sigma \odot \epsilon, \quad (= g(\epsilon, \phi, x))$$

for $\odot$ the element-wise product.
Obviously the Jacobian is $\mathrm{diag}(\sigma)$, so the 'log determinant' is $\sum_i \log(\sigma_i)$

$$\Rightarrow \log(q_\phi(z|x)) = \log(p(\epsilon)) - \log(|J_\phi(x, \epsilon)|)$$

$$= \sum_i \log(\mathcal{N}(\epsilon_i; 0, 1)) - \log(\sigma_i)$$

(2) for the full-covariance Gaussian posterior,

$$q_\phi(z|x) = \mathcal{N}(z; \mu, \Sigma)$$

The appropriate parametrisation is

$$\epsilon \sim \mathcal{N}(0, Id)$$

$$z = \mu + L\epsilon$$

for L some lower triangular matrix. We can parametrise this way since given A s.t. $AA^T = \Sigma$, a normal RV is $z = \mu + A\epsilon$.
But this A is not unique, and for any symmetric positive-definite matrix we can use the Cholesky decomposition to find a lower (or upper) triangular A.
i.e. L is

$$\Sigma = \mathbb{E}[(z - \mathbb{E}[z])(z - \mathbb{E}[z])^T]$$

$$= \mathbb{E}[(L\epsilon)(L\epsilon)^T]$$

$$= L\,\mathbb{E}[\epsilon\epsilon^T]\,L^T$$

$$= LL^T.$$

We can build such an L by having

$$(\mu, \log(\sigma), L') = \text{Encoder Neural Net}(x)$$

$$L = L_{mask} \odot L + \text{diag}(\sigma),$$

where $L_{mask} = \begin{pmatrix} 0 & \cdots & 0 \\ 1 & \ddots & \vdots \\ \vdots & \ddots & 0 \\ 1 & \cdots & 1 \; 0 \end{pmatrix}$, so 0 on and above the main diagonal.

So as before, the log-determinant is $\sum_i \log(\sigma_i)$.

- One can see that, in general, if $z = L\epsilon + \mu$ can be replaced with a chain of potentially non-linear transformations and provided the Jacobian is triangular with non-zero diagonal entries the log-determinant remains easy.

Estimation of the Marginal Likelihood

- After training a VAE, we can estimate the probability of data under the model. Obviously we have

$$\log(p_\theta(x)) = \log\left(\mathbb{E}_{z\sim q_\phi(z|x)}\left[\frac{p_\theta(x,z)}{q_\phi(z|x)}\right]\right)$$

$$\approx \log\left(\frac{1}{L}\sum_{\ell=1}^{L} p_\theta(x,z^\ell)/q_\phi(z^\ell|x)\right)$$

for z^ℓ i.i.d. $z^\ell \sim q_\phi(z,x)$

Generative Modelling

- We can now obviously draw random or conditional new samples from a learned distribution.
- For instance, to generate samples similar to x, do the following:

① Find the parameters ϕ = Neural Net (x) which parametrise q_ϕ.
i.e. use the encoder to get $q_\phi(z|x)$

② Sample z from this distribution.

③ Decode z to get
x_{new} = Decoder NeuralNet (z)

- Note that there is some subtlety in ③. Strictly, we learn parameters of $p(x|z)$ which outputs a mean function.
- We don't normally have a variance estimate here. This is because selecting the best point minimises reconstruction loss, in that if $p_\theta(x|z)$ is $N(\mu_\theta, \sigma_\theta^2 Id)$,

then $E_{q(z|x)}[-\log(p_\theta(x|z))]$ is ~~maximised~~ maximised by ~~[illegible]~~ minimising the squared error of x and $\mu_\theta(z)$.

- Note that really we could view ϕ as the encoding parameters and θ as the decoding (generative modelling) parameters, so for unconditional modelling we don't need ϕ at all! (Although obviously we still need it in order to learn p_θ).