Variational Autoencoder

Deep Learning (ENGG*6600*01)

School of Engineering,
University of Guelph, ON, Canada

Course Instructor: Benyamin Ghojogh
Summer 2023

Variational Inference

Variational Inference

- Consider a dataset $\{\boldsymbol{x}_i\}_{i=1}^n$. Assume that every data point $\boldsymbol{x}_i \in \mathbb{R}^d$ is generated from a latent variable $\boldsymbol{z}_i \in \mathbb{R}^p$. This latent variable has a prior distribution $\mathbb{P}(\boldsymbol{z}_i)$. According to Bayes' rule, we have:

$$\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i) = \frac{\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i)\, \mathbb{P}(\boldsymbol{z}_i)}{\mathbb{P}(\boldsymbol{x}_i)}. \tag{1}$$

- Let $\mathbb{P}(\boldsymbol{z}_i)$ be an arbitrary distribution denoted by $q(\boldsymbol{z}_i)$. Suppose the parameter of conditional distribution of $\boldsymbol{z}_i$ on $\boldsymbol{x}_i$ is denoted by $\boldsymbol{\theta}$; hence, $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i) = \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta})$. Therefore, we can say:

$$\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}) = \frac{\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i, \boldsymbol{\theta})\, \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{\theta})}{\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})}. \tag{2}$$

Variational Inference

- Consider the Kullback-Leibler (KL) divergence [1] between the prior probability of the latent variable and the posterior of the latent variable:

$$
\begin{aligned}
&\text{KL}\big(q(\boldsymbol{z}_i)\,\|\,\mathbb{P}(\boldsymbol{z}_i\,|\,\boldsymbol{x}_i,\boldsymbol{\theta})\big) \overset{(a)}{=} \int q(\boldsymbol{z}_i)\log\big(\frac{q(\boldsymbol{z}_i)}{\mathbb{P}(\boldsymbol{z}_i\,|\,\boldsymbol{x}_i,\boldsymbol{\theta})}\big)d\boldsymbol{z}_i \\
&= \int q(\boldsymbol{z}_i)\big(\log(q(\boldsymbol{z}_i)) - \log(\mathbb{P}(\boldsymbol{z}_i\,|\,\boldsymbol{x}_i,\boldsymbol{\theta}))\big)d\boldsymbol{z}_i \\
&\overset{(2)}{=} \int q(\boldsymbol{z}_i)\big(\log(q(\boldsymbol{z}_i)) - \log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{z}_i,\boldsymbol{\theta})) - \log(\mathbb{P}(\boldsymbol{z}_i\,|\,\boldsymbol{\theta})) + \log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{\theta}))\big)\,d\boldsymbol{z}_i \\
&\overset{(b)}{=} \log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{\theta})) + \int q(\boldsymbol{z}_i)\big(\log(q(\boldsymbol{z}_i)) - \log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{z}_i,\boldsymbol{\theta})) - \log(\mathbb{P}(\boldsymbol{z}_i\,|\,\boldsymbol{\theta}))\big)\,d\boldsymbol{z}_i \\
&= \log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{\theta})) + \int q(\boldsymbol{z}_i)\log\big(\frac{q(\boldsymbol{z}_i)}{\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{z}_i,\boldsymbol{\theta})\mathbb{P}(\boldsymbol{z}_i\,|\,\boldsymbol{\theta})}\big)\,d\boldsymbol{z}_i \\
&= \log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{\theta})) + \int q(\boldsymbol{z}_i)\log\big(\frac{q(\boldsymbol{z}_i)}{\mathbb{P}(\boldsymbol{x}_i,\boldsymbol{z}_i\,|\,\boldsymbol{\theta})}\big)\,d\boldsymbol{z}_i \\
&= \log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{\theta})) + \text{KL}\big(q(\boldsymbol{z}_i)\,\|\,\mathbb{P}(\boldsymbol{x}_i,\boldsymbol{z}_i\,|\,\boldsymbol{\theta})\big),
\end{aligned}
$$

  where (a) is for definition of KL divergence and (b) is because $\log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{\theta}))$ is independent of $\boldsymbol{z}_i$ and comes out of integral and $\int d\boldsymbol{z}_i = 1$.

- Hence:

$$
\log(\mathbb{P}(\boldsymbol{x}_i\,|\,\boldsymbol{\theta})) = \text{KL}\big(q(\boldsymbol{z}_i)\,\|\,\mathbb{P}(\boldsymbol{z}_i\,|\,\boldsymbol{x}_i,\boldsymbol{\theta})\big) - \text{KL}\big(q(\boldsymbol{z}_i)\,\|\,\mathbb{P}(\boldsymbol{x}_i,\boldsymbol{z}_i\,|\,\boldsymbol{\theta})\big). \quad (3)
$$

Variational Inference

- We found:

$$\log(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})) = \text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta})\big) - \text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big).$$

- We define the **Evidence Lower Bound (ELBO)** as:

$$\mathcal{L}(q, \boldsymbol{\theta}) := -\text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big). \tag{4}$$

  So:

$$\log(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})) = \text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta})\big) + \mathcal{L}(q, \boldsymbol{\theta}).$$

- Therefore:

$$\mathcal{L}(q, \boldsymbol{\theta}) = \log(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})) - \underbrace{\text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta})\big)}_{\geq 0}. \tag{5}$$

- As the second term is negative with its minus, the ELBO is a lower bound on the log likelihood of data:

$$\mathcal{L}(q, \boldsymbol{\theta}) \leq \log(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})). \tag{6}$$

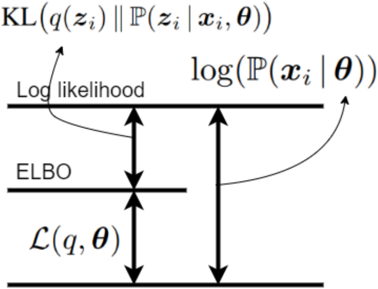
  The likelihood $\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})$ is also referred to as the **evidence**.
- Note that this lower bound gets tight when:

$$\mathcal{L}(q, \boldsymbol{\theta}) \approx \log(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})) \implies 0 \leq \text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta})\big) \overset{\text{set}}{=} 0$$
$$\implies q(\boldsymbol{z}_i) = \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}). \tag{7}$$

Variational Inference

- We found:

$$\log(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})) = \text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta})\big) + \mathcal{L}(q, \boldsymbol{\theta}).$$



Expectation Maximization in Variational Inference

- According to MLE, we want to maximize the log-likelihood of data. According to Eq. (6):

$$\mathcal{L}(q, \boldsymbol{\theta}) \leq \log(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{\theta})),$$

maximizing the ELBO will also maximize the log-likelihood.

- The Eq. (6) holds for any prior distribution q. We want to find the best distribution to maximize the lower bound.
- Hence, EM for variational inference is performed iteratively as:

$$\text{E-step:} \quad q^{(t)} := \arg\max_{q} \quad \mathcal{L}(q, \boldsymbol{\theta}^{(t-1)}), \tag{8}$$

$$\text{M-step:} \quad \boldsymbol{\theta}^{(t)} := \arg\max_{\boldsymbol{\theta}} \quad \mathcal{L}(q^{(t)}, \boldsymbol{\theta}), \tag{9}$$

where t denotes the iteration index.

Expectation Maximization in Variational Inference

- **E-step in EM for Variational Inference**: The E-step is:

$$\max_q \mathcal{L}(q, \boldsymbol{\theta}^{(t-1)}) \overset{(5)}{=} \max_q \log(\mathbb{P}(\boldsymbol{x}_i \mid \boldsymbol{\theta}^{(t-1)})) + \max_q \big(-\text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}^{(t-1)})\big)\big)$$

$$= \max_q \log(\mathbb{P}(\boldsymbol{x}_i \mid \boldsymbol{\theta}^{(t-1)})) + \min_q \text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}^{(t-1)})\big).$$

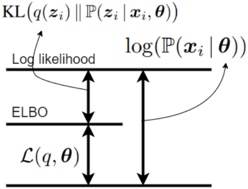
- The second term is always non-negative; hence, its minimum is zero:

$$\text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{z}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}^{(t-1)})\big) \overset{\text{set}}{=} 0 \implies q(\boldsymbol{z}_i) = \mathbb{P}(\boldsymbol{z}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}^{(t-1)}),$$

which was already found in Eq. (7). Thus, the E-step assigns:

$$q^{(t)}(\boldsymbol{z}_i) \leftarrow \mathbb{P}(\boldsymbol{z}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}^{(t-1)}). \tag{10}$$

- In other words, in the figure, it pushes the middle line toward the above line by maximizing the ELBO.



Expectation Maximization in Variational Inference

- **M-step in EM for Variational Inference**: The M-step is:

$$\max_{\boldsymbol{\theta}} \mathcal{L}(q^{(t)}, \boldsymbol{\theta}) \overset{(4)}{=} \max_{\boldsymbol{\theta}} \big(-\text{KL}\big(q^{(t)}(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big)\big)$$

$$\overset{(a)}{=} \max_{\boldsymbol{\theta}} \Big[-\int q^{(t)}(\boldsymbol{z}_i) \log\Big(\frac{q^{(t)}(\boldsymbol{z}_i)}{\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})}\Big)\, d\boldsymbol{z}_i\Big]$$

$$= \max_{\boldsymbol{\theta}} \int q^{(t)}(\boldsymbol{z}_i) \log(\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}))\, d\boldsymbol{z}_i - \max_{\boldsymbol{\theta}} \int q^{(t)}(\boldsymbol{z}_i) \log(q^{(t)}(\boldsymbol{z}_i))\, d\boldsymbol{z}_i,$$

  where (a) is for definition of KL divergence.

- The second term is constant w.r.t. $\boldsymbol{\theta}$. Hence:

$$\max_{\boldsymbol{\theta}} \mathcal{L}(q^{(t)}, \boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \int q^{(t)}(\boldsymbol{z}_i) \log(\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}))\, d\boldsymbol{z}_i$$

$$\overset{(a)}{=} \max_{\boldsymbol{\theta}} \mathbb{E}_{\sim q^{(t)}(\boldsymbol{z}_i)}\big[\log \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big],$$

  where (a) is because of definition of expectation. Thus, the M-step assigns:

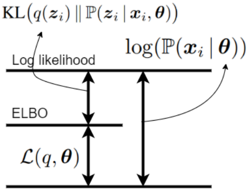
$$\boldsymbol{\theta}^{(t)} \leftarrow \arg\max_{\boldsymbol{\theta}} \; \mathbb{E}_{\sim q^{(t)}(\boldsymbol{z}_i)}\big[\log \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big]. \tag{11}$$

Expectation Maximization in Variational Inference

- We found:

$$\boldsymbol{\theta}^{(t)} \leftarrow \arg\max_{\boldsymbol{\theta}} \; \mathbb{E}_{\sim q^{(t)}(\boldsymbol{z}_i)}\big[\log \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big].$$

- In other words, in the figure, it pushes the above line higher.



- The E-step and M-step together somehow play a **game** where the E-step tries to reach the middle line (or the ELBO) to the log-likelihood and the M-step tries to increase the above line (or the log-likelihood). This procedure is done repeatedly so the two steps help each other improve to higher values.
- To summarize, the EM in variational inference is:

$$q^{(t)}(\boldsymbol{z}_i) \leftarrow \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}^{(t-1)}), \tag{12}$$

$$\boldsymbol{\theta}^{(t)} \leftarrow \arg\max_{\boldsymbol{\theta}} \; \mathbb{E}_{\sim q^{(t)}(\boldsymbol{z}_i)}\big[\log \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big]. \tag{13}$$

Expectation Maximization in Variational Inference

- It is noteworthy that, in variational inference, sometimes, the parameter $\boldsymbol{\theta}$ is absorbed into the latent variable $\boldsymbol{z}_i$.
- According to the chain rule, we have:

$$\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i, \boldsymbol{\theta}) = \mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i, \boldsymbol{\theta})\, \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{\theta})\, \mathbb{P}(\boldsymbol{\theta}).$$

- Considering the term $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{\theta})\, \mathbb{P}(\boldsymbol{\theta})$ as one probability term, we have:

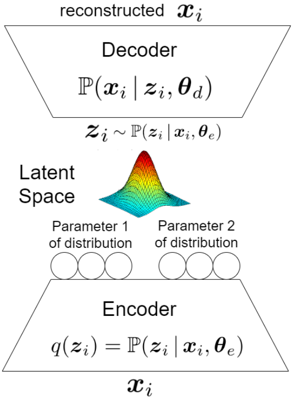
$$\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i) = \mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i)\, \mathbb{P}(\boldsymbol{z}_i),$$

  where the parameter $\boldsymbol{\theta}$ disappears because of absorption.

Variational Autoencoder

Variational Autoencoder

- **Variational Autoencoder (VAE)** (2014) [2] applies variational inference, i.e., maximizes the ELBO, but in an **autoencoder** setup and makes it differentiable for the **backpropagation** training [3].
- As this figure shows, VAE includes an **encoder** and a **decoder**, each of which can have several network layers. A **latent space** is learned between the encoder and decoder. The **latent variable** $\boldsymbol{z}_i$ is **sampled** from the latent space. The input of encoder in VAE is the **data point** $\boldsymbol{x}_i$ and the output of decoder in VAE is its **reconstruction** $\boldsymbol{x}_i$.



Encoder of Variational Autoencoder

- The encoder of VAE models the distribution $q(\boldsymbol{z}_i) = \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$ where the parameters of distribution $\boldsymbol{\theta}_e$ are the weights of encoder layers in VAE.
- The input and output of encoder are $\boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{z}_i \in \mathbb{R}^p$, respectively.
- As the figure depicts, the output neurons of encoder are supposed to determine the parameters of the conditional distribution $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$. If this conditional distribution has m **number of parameters**, we have m **sets of output neurons from the encoder**, denoted by $\{\boldsymbol{e}_j\}_{j=1}^m$. The dimensionality of these sets may differ depending on the size of the parameters.
- For example, let the latent space be p-dimensional, i.e., $\boldsymbol{z}_i \in \mathbb{R}^p$. If the distribution $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$ is a **multivariate Gaussian distribution**, we have two sets of output neurons for encoder where one set has p neurons for the **mean** of this distribution $\boldsymbol{\mu}_{z|x} = \boldsymbol{e}_1 \in \mathbb{R}^p$ and the other set has $(p \times p)$ neurons for the **covariance** of this distribution $\boldsymbol{\Sigma}_{z|x} =$ matrix form of $\boldsymbol{e}_2 \in \mathbb{R}^{p \times p}$. If the covariance matrix is **diagonal**, the second set has p neurons rather than $(p \times p)$ neurons. In this case, we have $\boldsymbol{\Sigma}_{z|x} = \textbf{diag}(\boldsymbol{e}_2) \in \mathbb{R}^{p \times p}$.
- Any distribution with any number of parameters can be chosen for $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$ but the **multivariate Gaussian with diagonal covariance** is very well-used:

$$q(\boldsymbol{z}_i) = \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) = \mathcal{N}(\boldsymbol{z}_i \,|\, \boldsymbol{\mu}_{z|x}, \boldsymbol{\Sigma}_{z|x}). \tag{14}$$

- Let the network **weights** for the output sets of encoder, $\{\boldsymbol{e}_j\}_{j=1}^m$, be denoted by $\{\boldsymbol{\theta}_{e,j}\}_{j=1}^m$. As the input of encoder is $\boldsymbol{x}_i$, the j-th output set of encoder can be written as $\boldsymbol{e}_j(\boldsymbol{x}_i, \boldsymbol{\theta}_{e,j})$. In the case of multivariate Gaussian distribution for the latent space, the parameters are $\boldsymbol{\mu}_{z|x} = \boldsymbol{e}_1(\boldsymbol{x}_i, \boldsymbol{\theta}_{e,1})$ and $\boldsymbol{\Sigma}_{z|x} = \textbf{diag}(\boldsymbol{e}_2(\boldsymbol{x}_i, \boldsymbol{\theta}_{e,2}))$.

Sampling the Latent Variable

- When the **data point $\boldsymbol{x}_i$ is fed as input** to the encoder, the parameters of the conditional distribution $q(\boldsymbol{z}_i)$ are obtained; hence, the **distribution of latent space, which is $q(\boldsymbol{z}_i)$, is determined corresponding to the data point $\boldsymbol{x}_i$**.
- Now, in the latent space, we **sample** the corresponding latent variable from the distribution of latent space:

$$\boldsymbol{z}_i \sim q(\boldsymbol{z}_i) = \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e). \tag{15}$$

- This latent variable is **fed as input to the decoder** which is explained in the following.

Decoder of Variational Autoencoder

- As the figure shows, the decoder of VAE models the conditional distribution $\mathbb{P}(\boldsymbol{x}_i \mid \boldsymbol{z}_i, \boldsymbol{\theta}_d)$ where $\boldsymbol{\theta}_d$ are the weights of decoder layers in VAE.
- The input and output of decoder are $\boldsymbol{z}_i \in \mathbb{R}^p$ and $\boldsymbol{x}_i \in \mathbb{R}^d$, respectively. The output neurons of decoder are supposed to either generate the reconstructed data point or determine the parameters of the conditional distribution $\mathbb{P}(\boldsymbol{x}_i \mid \boldsymbol{z}_i, \boldsymbol{\theta}_d)$.
- The former is more common.
- In the latter case, if this conditional distribution has l number of parameters, we have l sets of output neurons from the decoder, denoted by $\{\boldsymbol{d}_j\}_{j=1}^l$. The dimensionality of these sets may differ depending the size of every parameters. The example of multivariate Gaussian distribution also can be mentioned for the decoder.
- Let the network **weights** for the output sets of decoder, $\{\boldsymbol{d}_j\}_{j=1}^l$, be denoted by $\{\boldsymbol{\theta}_{d,j}\}_{j=1}^l$. As the input of decoder is $\boldsymbol{z}_i$, the j-th output set of decoder can be written as $\boldsymbol{d}_j(\boldsymbol{z}_i, \boldsymbol{\theta}_{d,j})$.

Training Variational Autoencoder with Expectation Maximization

- We use EM for training the VAE. Recall Eqs. (8) and (9) for EM in variational inference:

$$\text{E-step:} \quad q^{(t)} := \arg\max_{q} \quad \mathcal{L}(q, \boldsymbol{\theta}^{(t-1)}),$$

$$\text{M-step:} \quad \boldsymbol{\theta}^{(t)} := \arg\max_{\boldsymbol{\theta}} \quad \mathcal{L}(q^{(t)}, \boldsymbol{\theta}).$$

- Inspired by that, VAE uses EM for training where the ELBO is a function of encoder weights $\boldsymbol{\theta}_e$, decoder weights $\boldsymbol{\theta}_d$, and data point $\boldsymbol{x}_i$:

$$\text{E-step:} \quad \boldsymbol{\theta}_e^{(t)} := \arg\max_{q} \quad \mathcal{L}(\boldsymbol{\theta}_e, \boldsymbol{\theta}_d^{(t-1)}, \boldsymbol{x}_i), \tag{16}$$

$$\text{M-step:} \quad \boldsymbol{\theta}_d^{(t)} := \arg\max_{q} \quad \mathcal{L}(\boldsymbol{\theta}_e^{(t)}, \boldsymbol{\theta}_d, \boldsymbol{x}_i). \tag{17}$$

Training Variational Autoencoder with Expectation Maximization

- We had:

$$\text{E-step:} \quad \boldsymbol{\theta}_e^{(t)} := \arg\max_q \quad \mathcal{L}(\boldsymbol{\theta}_e, \boldsymbol{\theta}_d^{(t-1)}, \boldsymbol{x}_i),$$
$$\text{M-step:} \quad \boldsymbol{\theta}_d^{(t)} := \arg\max_q \quad \mathcal{L}(\boldsymbol{\theta}_e^{(t)}, \boldsymbol{\theta}_d, \boldsymbol{x}_i).$$

- We can simplify this iterative optimization algorithm by **alternating optimization** [4] where we take a step of gradient ascent optimization in every iteration. We consider mini-batch stochastic gradient ascent and take training data in batches where b denotes the mini-batch size. Hence, the optimization is:

$$\text{E-step:} \quad \boldsymbol{\theta}_e^{(t)} := \boldsymbol{\theta}_e^{(t-1)} + \eta_e \frac{\partial \sum_{i=1}^{b} \mathcal{L}(\boldsymbol{\theta}_e, \boldsymbol{\theta}_d^{(t-1)}, \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_e}, \tag{18}$$
$$\text{M-step:} \quad \boldsymbol{\theta}_d^{(t)} := \boldsymbol{\theta}_d^{(t-1)} + \eta_d \frac{\partial \sum_{i=1}^{b} \mathcal{L}(\boldsymbol{\theta}_e^{(t)}, \boldsymbol{\theta}_d, \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_d}, \tag{19}$$

where η_e and η_d are the learning rates for $\boldsymbol{\theta}_e$ and $\boldsymbol{\theta}_d$, respectively.

Training Variational Autoencoder with Expectation Maximization

- Eqs. (4) and (12) were:

$$\mathcal{L}(q, \boldsymbol{\theta}) := -\text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta})\big),$$
$$q^{(t)}(\boldsymbol{z}_i) \leftarrow \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}^{(t-1)}).$$

- The ELBO is simplified as:

$$\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) \overset{(4)}{=} -\sum_{i=1}^{b} \text{KL}\big(q(\boldsymbol{z}_i) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)\big)$$
$$\overset{(12)}{=} -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)\big). \quad (20)$$

- Note that the parameter of $\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)$ is $\boldsymbol{\theta}_d$ because $\boldsymbol{z}_i$ is generated after the encoder and before the decoder.
- There are different ways for **approximating the KL divergence** in Eq. (20) [5, 6]. We can simplify the ELBO in at least two different ways which are explained in the following.

Simplification Type 1

- We continue the simplification of ELBO:

$$\begin{aligned}\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) &= -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)\big) \\ &= -\sum_{i=1}^{b} \mathbb{E}_{\sim q^{(t-1)}(\boldsymbol{z}_i)}\Big[\log\big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)}\big)\Big] \\ &= -\sum_{i=1}^{b} \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}\Big[\log\big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)}\big)\Big]. \qquad (21)\end{aligned}$$

- This expectation can be approximated using **Monte Carlo approximation** [7] where we draw ℓ samples $\{\boldsymbol{z}_{i,j}\}_{j=1}^{\ell}$, corresponding to the i-th data point, from the conditional distribution distribution as:

$$\boldsymbol{z}_{i,j} \sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e), \quad \forall j \in \{1, \dots, \ell\}. \qquad (22)$$

- Monte Carlo approximation [7], in general, approximates expectation as:

$$\mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}\big[f(\boldsymbol{z}_i)\big] \approx \frac{1}{\ell} \sum_{j=1}^{\ell} f(\boldsymbol{z}_{i,j}), \qquad (23)$$

where $f(\boldsymbol{z}_i)$ is a function of $\boldsymbol{z}_i$.

Simplification Type 1

- We had:

$$\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) = -\sum_{i=1}^{b} \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)} \Big[\log \big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)} \big) \Big].$$

- Here, the approximation is:

$$\begin{aligned} \sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) &\approx \sum_{i=1}^{b} \widetilde{\mathcal{L}}(q, \boldsymbol{\theta}) \\ &= -\sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \log \big(\frac{\mathbb{P}(\boldsymbol{z}_{i,j} \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_{i,j} \,|\, \boldsymbol{\theta}_d)} \big) \\ &= \sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \Big[\log \big(\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_{i,j} \,|\, \boldsymbol{\theta}_d) \big) - \log \big(\mathbb{P}(\boldsymbol{z}_{i,j} \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \big) \Big]. \end{aligned} \tag{24}$$

Simplification Type 2

- We can simplify the ELBO using another approach:

$$
\begin{aligned}
\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) &= -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)\big) \\
&= -\sum_{i=1}^{b} \int \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \log\big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)}\big)\, d\boldsymbol{z}_i \\
&= -\sum_{i=1}^{b} \int \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \log\big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i, \boldsymbol{\theta}_d)\, \mathbb{P}(\boldsymbol{z}_i)}\big)\, d\boldsymbol{z}_i \\
&= -\sum_{i=1}^{b} \int \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \log\big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{z}_i)}\big)\, d\boldsymbol{z}_i \\
&\quad + \sum_{i=1}^{b} \int \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i, \boldsymbol{\theta}_d)\big)\, d\boldsymbol{z}_i \\
&= -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{z}_i)\big) \\
&\quad + \sum_{i=1}^{b} \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}\Big[\log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i, \boldsymbol{\theta}_d)\big)\Big]. \qquad (25)
\end{aligned}
$$

Simplification Type 2

- We found:

$$\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) = -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{z}_i)\big) + \sum_{i=1}^{b} \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}\Big[\log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i, \boldsymbol{\theta}_d)\big)\Big].$$

- The second term in the above equation can be estimated using **Monte Carlo approximation** [7] where we draw ℓ samples $\{\boldsymbol{z}_{i,j}\}_{j=1}^{\ell}$ from $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$:

$$\begin{aligned}\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) &\approx \sum_{i=1}^{b} \widetilde{\mathcal{L}}(q, \boldsymbol{\theta}) \\ &= -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{z}_i)\big) + \sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_{i,j}, \boldsymbol{\theta}_d)\big). \end{aligned} \tag{26}$$

Simplification Type 2

- We had:

$$\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) \approx -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{z}_i)\big) + \sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_{i,j}, \boldsymbol{\theta}_d)\big).$$

- The first term in the above equation can be **converted to expectation** and then computed using **Monte Monte Carlo approximation** [7] again, where we draw ℓ samples $\{\boldsymbol{z}_{i,j}\}_{j=1}^{\ell}$ from $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$:

$$\begin{aligned} \sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) &\approx \sum_{i=1}^{b} \widetilde{\mathcal{L}}(q, \boldsymbol{\theta}) \\ &= -\sum_{i=1}^{b} \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)} \Big[\log\Big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{z}_i)}\Big)\Big] + \sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_{i,j}, \boldsymbol{\theta}_d)\big) \\ &\approx -\sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \log\big(\mathbb{P}(\boldsymbol{z}_{i,j} \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)\big) - \log\big(\mathbb{P}(\boldsymbol{z}_{i,j})\big) + \sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_{i,j}, \boldsymbol{\theta}_d)\big). \end{aligned} \tag{27}$$

- In case we have some families of distributions, such as **Gaussian** distributions, for $\mathbb{P}(\boldsymbol{z}_{i,j} \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$ and $\mathbb{P}(\boldsymbol{z}_{i,j})$, the first term in Eq. (26) can be **computed analytically**. In the following, we simply Eq. (26) further for Gaussian distributions.

Simplification Type 2 for Special Case of Gaussian Distributions

- We can compute the KL divergence in the **first term** of Eq. (26) analytically for **univariate or multivariate Gaussian** distributions. For this, we need two following lemmas (see our tutorial paper [8] for proof).

Lemma

The KL divergence between two univariate Gaussian distributions $p_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$ *and* $p_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$ *is:*

$$KL(p_1 \| p_2) = \log(\frac{\sigma_2}{\sigma_1}) + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}. \tag{28}$$

Lemma

The KL divergence between two multivariate Gaussian distributions $p_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ *and* $p_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ *with dimensionality* p *is:*

$$KL(p_1 \| p_2) = \frac{1}{2}\Big(\log(\frac{|\boldsymbol{\Sigma}_2|}{|\boldsymbol{\Sigma}_1|}) - p + \mathbf{tr}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1) + (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^\top \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)\Big). \tag{29}$$

Simplification Type 2 for Special Case of Gaussian Distributions

- Consider the case in which we have:

$$\mathbb{P}(\boldsymbol{z}_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}_e) \sim \mathcal{N}(\boldsymbol{\mu}_{z|x}, \boldsymbol{\Sigma}_{z|x}), \tag{30}$$

$$\mathbb{P}(\boldsymbol{z}_i) \sim \mathcal{N}(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z), \tag{31}$$

  where $\boldsymbol{z}_i \in \mathbb{R}^p$. Note that the parameters $\boldsymbol{\mu}_{z|x}$ and $\boldsymbol{\Sigma}_{z|x}$ are trained in neural network while the parameters $\mathbb{P}(\boldsymbol{z}_{i,j})$ can be set to $\boldsymbol{\mu}_z = \boldsymbol{0}$ and $\boldsymbol{\Sigma}_z = \boldsymbol{I}$ (inspired by the prior distribution of $\boldsymbol{z}$ in factor analysis).

- According to Lemma 2, the approximation of ELBO, i.e. Eq. (26), can be simplified to:

$$\begin{aligned}
&\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) \approx \sum_{i=1}^{b} \widetilde{\mathcal{L}}(q, \boldsymbol{\theta}) \\
&= -\sum_{i=1}^{b} \frac{1}{2}\Big(\log(\frac{|\boldsymbol{\Sigma}_z|}{|\boldsymbol{\Sigma}_{z|x}|}) - p + \mathbf{tr}(\boldsymbol{\Sigma}_z^{-1}\boldsymbol{\Sigma}_{z|x}) + (\boldsymbol{\mu}_z - \boldsymbol{\mu}_{z|x})^\top \boldsymbol{\Sigma}_z^{-1}(\boldsymbol{\mu}_z - \boldsymbol{\mu}_{z|x})\Big) \\
&\quad + \sum_{i=1}^{b} \frac{1}{\ell} \sum_{j=1}^{\ell} \log\big(\mathbb{P}(\boldsymbol{x}_i \mid \boldsymbol{z}_{i,j}, \boldsymbol{\theta}_d)\big).
\end{aligned} \tag{32}$$

Training Variational Autoencoder with Approximations

- We can train VAE with EM, where Monte Carlo approximations are applied to ELBO. The Eqs. (18) and (19):

$$\text{E-step:} \quad \boldsymbol{\theta}_e^{(t)} := \boldsymbol{\theta}_e^{(t-1)} + \eta_e \frac{\partial \sum_{i=1}^{b} \mathcal{L}(\boldsymbol{\theta}_e, \boldsymbol{\theta}_d^{(t-1)}, \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_e},$$

$$\text{M-step:} \quad \boldsymbol{\theta}_d^{(t)} := \boldsymbol{\theta}_d^{(t-1)} + \eta_d \frac{\partial \sum_{i=1}^{b} \mathcal{L}(\boldsymbol{\theta}_e^{(t)}, \boldsymbol{\theta}_d, \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_d},$$

are replaced by the following equations:

$$\text{E-step:} \quad \boldsymbol{\theta}_e^{(t)} := \boldsymbol{\theta}_e^{(t-1)} + \eta_e \frac{\partial \sum_{i=1}^{b} \widetilde{\mathcal{L}}(\boldsymbol{\theta}_e, \boldsymbol{\theta}_d^{(t-1)}, \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_e}, \tag{33}$$

$$\text{M-step:} \quad \boldsymbol{\theta}_d^{(t)} := \boldsymbol{\theta}_d^{(t-1)} + \eta_d \frac{\partial \sum_{i=1}^{b} \widetilde{\mathcal{L}}(\boldsymbol{\theta}_e^{(t)}, \boldsymbol{\theta}_d, \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}_d}, \tag{34}$$

where the approximated ELBO was introduced in previous sections.

The Reparameterization Trick

- Sampling the ℓ samples for the latent variables, i.e. Eq. (15):

$$\boldsymbol{z}_i \sim q(\boldsymbol{z}_i) = \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e),$$

  blocks the gradient flow because computing the **derivatives** through $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$ by chain rule gives a **high variance estimate of gradient**.

- In order to overcome this problem, we use the **reparameterization technique** (2014) [2, 9, 10]. In this technique, instead of sampling $\boldsymbol{z}_i \sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$, we assume $\boldsymbol{z}_i$ is a random variable but is a **deterministic function of another random variable** $\boldsymbol{\epsilon}_i$ as follows:

$$\boldsymbol{z}_i = g(\boldsymbol{\epsilon}_i, \boldsymbol{x}_i, \boldsymbol{\theta}_e), \tag{35}$$

  where $\boldsymbol{\epsilon}_i$ is a stochastic variable sampled from a distribution as:

$$\boldsymbol{\epsilon}_i \sim \mathbb{P}(\boldsymbol{\epsilon}). \tag{36}$$

The Reparameterization Trick

- The Eqs. (21) and (25):

$$\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) = -\sum_{i=1}^{b} \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}\Big[\log\big(\frac{\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}{\mathbb{P}(\boldsymbol{x}_i, \boldsymbol{z}_i \,|\, \boldsymbol{\theta}_d)}\big)\Big],$$

$$\sum_{i=1}^{b} \mathcal{L}(q, \boldsymbol{\theta}) = -\sum_{i=1}^{b} \text{KL}\big(\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e) \,\|\, \mathbb{P}(\boldsymbol{z}_i)\big) + \sum_{i=1}^{b} \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}\Big[\log\big(\mathbb{P}(\boldsymbol{x}_i \,|\, \boldsymbol{z}_i, \boldsymbol{\theta}_d)\big)\Big],$$

  both contain an expectation of a function $f(\boldsymbol{z}_i)$. Using this technique, this expectation is replaced as:

$$\mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}[f(\boldsymbol{z}_i)] \rightarrow \mathbb{E}_{\sim \mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)}[f(g(\boldsymbol{\epsilon}_i, \boldsymbol{x}_i, \boldsymbol{\theta}_e))]. \tag{37}$$

- Using the reparameterization technique, the **encoder**, which implemented $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$, is replaced by $g(\boldsymbol{\epsilon}_i, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$ where in the latent space between encoder and decoder, we have $\boldsymbol{\epsilon}_i \sim \mathbb{P}(\boldsymbol{\epsilon})$ and $\boldsymbol{z}_i = g(\boldsymbol{\epsilon}_i, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$.
- A simple example for the reparameterization technique is when z_i and ϵ_i are univariate Gaussian variables:

$$z_i \sim \mathcal{N}(\mu, \sigma^2),$$
$$\epsilon_i \sim \mathcal{N}(0, 1),$$
$$z_i = g(\epsilon_i) = \mu + \sigma \epsilon_i.$$

- For some more advanced reparameterization techniques, the reader can refer to [11].

Training Variational Autoencoder with Backpropagation

- In practice, VAE is trained by **backpropagation** [9] where the backpropagation algorithm [3] is used for training the weights of network.
- Recall that in **training VAE with EM**, the **encoder and decoder are trained separately using the E-step and the M-step of EM, respectively**.
- However, in **training VAE with backpropagation**, the **whole network is trained together and not in separate steps**.
- Suppose the whole weights of VAE are denoted by $\boldsymbol{\theta} := \{\boldsymbol{\theta}_e, \boldsymbol{\theta}_d\}$. Backpropagation trains VAE using the **mini-batch stochastic gradient descent** with the **negative ELBO**, $\sum_{i=1}^{b} -\widetilde{\mathcal{L}}(\boldsymbol{\theta}, \boldsymbol{x}_i)$, as the loss function:

$$\boldsymbol{\theta}^{(t)} := \boldsymbol{\theta}^{(t-1)} - \eta \frac{\partial \sum_{i=1}^{b} -\widetilde{\mathcal{L}}(\boldsymbol{\theta}, \boldsymbol{x}_i)}{\partial \boldsymbol{\theta}}, \tag{38}$$

  where η is the learning rate. Note that we are **minimizing** here because neural networks usually minimize the loss function.

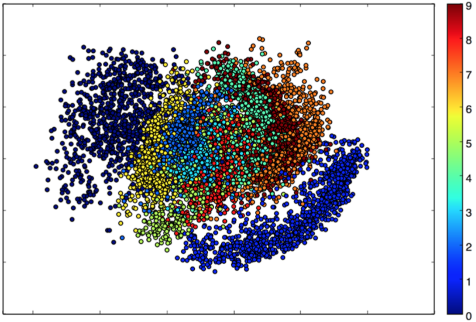
The Test Phase in Variational Autoencoder

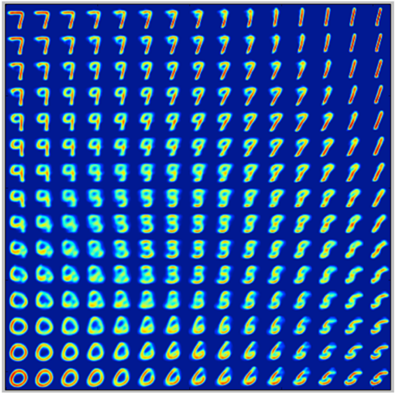
- In the test phase, we **feed the test data point $\boldsymbol{x}_i$ to the encoder** to determine the **parameters of the conditional distribution of latent space**, i.e., $\mathbb{P}(\boldsymbol{z}_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}_e)$.
- Then, from this distribution, we **sample** the latent variable $\boldsymbol{z}_i$ from the latent space and generate the corresponding **reconstructed data point $\boldsymbol{x}_i$** by the **decoder**.
- As you see, VAE is a **generative model** which generates data points [12].

Blurry Images Generated by VAE

- One of the problems of VAE is generating **blurry images** when data points are images. This blurry artifact may be because of several following reasons:
  - sampling for the **Monte Carlo approximations**
  - **lower bound approximation** by ELBO
  - **restrictions on the family of distributions** where usually simple Gaussian distributions are used.
- Note that **generative adversarial networks** [13] usually generate clearer images; therefore, some works have combined variational and adversarial inferences [14] for using the advantages of both models.

Simulation on MNIST Digit Dataset





Credit of image: `https://blog.keras.io/building-autoencoders-in-keras.html`

Acknowledgment

- Some slides are based on our tutorial paper: "Factor analysis, probabilistic principal component analysis, variational inference, and variational autoencoder: Tutorial and survey" [8]
- Some slides of this slide deck are inspired by teachings of deep learning course at the Carnegie Mellon University (you can see their YouTube channel).
- Variational autoencoder in Keras:
  - `https://blog.keras.io/building-autoencoders-in-keras.html`
  - `https://keras.io/examples/generative/vae/`

References

[1] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[2] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *International Conference on Learning Representations*, 2014.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[4] P. Jain and P. Kar, "Non-convex optimization for machine learning," *Foundations and Trends® in Machine Learning*, vol. 10, no. 3-4, pp. 142–336, 2017.

[5] J. R. Hershey and P. A. Olsen, "Approximating the Kullback Leibler divergence between Gaussian mixture models," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 4, pp. IV–317, IEEE, 2007.

[6] J. Duchi, "Derivations for linear algebra and optimization," tech. rep., Berkeley, California, 2007.

[7] B. Ghojogh, H. Nekoei, A. Ghojogh, F. Karray, and M. Crowley, "Sampling algorithms, from survey sampling to Monte Carlo methods: Tutorial and literature review," *arXiv preprint arXiv:2011.00901*, 2020.

[8] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, "Factor analysis, probabilistic principal component analysis, variational inference, and variational autoencoder: Tutorial and survey," *arXiv preprint arXiv:2101.00734*, 2021.

References (cont.)

[9] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *International Conference on Machine Learning*, 2014.

[10] M. Titsias and M. Lázaro-Gredilla, "Doubly stochastic variational Bayes for non-conjugate inference," in *International conference on machine learning*, pp. 1971–1979, 2014.

[11] M. Figurnov, S. Mohamed, and A. Mnih, "Implicit reparameterization gradients," *Advances in Neural Information Processing Systems*, vol. 31, pp. 441–452, 2018.

[12] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes," in *Advances in neural information processing systems*, pp. 841–848, 2002.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[14] L. Mescheder, S. Nowozin, and A. Geiger, "Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks," in *International Conference on Machine Learning*, 2017.