Latent Variable Models

Han Cheol Moon
School of Computer Science and Engineering
Nanyang Technological University
Singapore
hancheol001@edu.ntu.edu.sg

March 10, 2023

Contents

Chapter 1

Introduction

1.1 Introduction

1.1.1 Motivation of Latent Variable Models

If we knew a corresponding latent variable for each observation, then modelling might be easier. Imagine, how can we find $z^* =_z p(\mathbf{x}|\mathbf{z})$ for the data $\mathbf{x}$ as shown in Fig. 1.1(b)

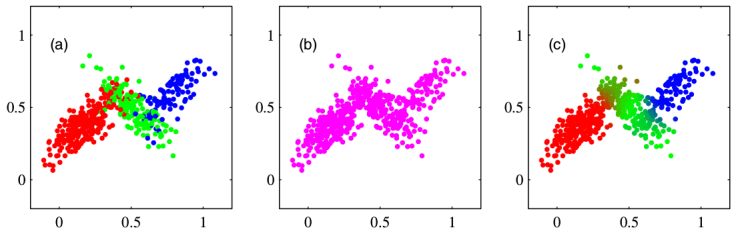

Figure 1.1: (a) Complete data set $p(\mathbf{x}|\mathbf{z})$. (b) Incomplete data set $p(\mathbf{x})$. (c) Inference result

For example, we want to model the complete data set $p(\mathbf{x}|\mathbf{z})$ under the i.i.d. assumption

$$p(\mathbf{x}_i, \mathbf{z}_i|\boldsymbol{\theta}) = \begin{cases} p(\mathcal{C}_1)p(\mathbf{x}_i|\mathcal{C}_1) \text{ if } z_i = 0 \\ p(\mathcal{C}_2)p(\mathbf{x}_i|\mathcal{C}_2) \text{ if } z_i = 1 \\ p(\mathcal{C}_3)p(\mathbf{x}_i|\mathcal{C}_3) \text{ if } z_i = 2 \end{cases}$$

$$p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N, \mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N|\boldsymbol{\theta}) = \prod_{n=1}^{N}\prod_{k=1}^{K} \pi_k^{z_{nk}} \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}}$$

, where $\pi_k = p(\mathcal{C}_k)$ and $p(\mathbf{x}_i|\mathcal{C}_k) = \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. However, in many cases, it is not observable.

1.2 K-Means Clustering

Suppose we have a data set $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ consisting of N observations of a random D-dimensional variable $\mathbf{x} \in R^D$. Our goal is to partition the data into some number K of clusters. Intuitively, we may think of a cluster as comprising a group of data points whose inter-point distances are small compared with the distances to points outside of the cluster.

This notion can be formalized by introducing a set of D-dimensional vectors $\boldsymbol{\mu}_k$, which represents the centers of the clusters. Our goal is to find an assignment of data points to clusters, as well as a set of vectors $\{\boldsymbol{\mu}_k\}$. Objective function of K-means clustering (distortion measure) can be defined as follows:

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||\boldsymbol{x}_n - \boldsymbol{\mu}_k||^2$$

, where $r_{nk} \in \{0, 1\}$ is a binary indicator variable which represents the membership of data $\mathbf{x}_n$. It can be defined as follows:

$$r_{nk} = \begin{cases} 1 & \text{if } k =_j ||\boldsymbol{x}_n - \boldsymbol{\mu}_j||^2 \\ 0 & \text{otherwise} \end{cases}$$

Subsequently, fix the cluster, and optimize the objective function with $\boldsymbol{\mu}_k$

1. $$2 \sum_{n=1}^{N} r_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k) = 0$$

2. $$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk} \boldsymbol{x}_n}{\sum_n r_{nk}}$$

The denominator of $\boldsymbol{\mu}_k$ is equal to the number of points assigned to cluster k. The mean of cluster k is essentially the same as the mean of data points $\mathbf{x}_n$ assigned to cluster k. For this reason, the procedure is known as the K-means clustering algorithm.

The two pahses of re-assigning data points to clusters and re-computing the cluster means are repeated in turn until there is no further change in the assignments. These two pahses reduce the value of the objective function J, so the convergence of the algorithm is assured. However, it may converge to a local rather than global minimum of J.

1.3 Gaussian Mixture Models

K-means clustering is a hard-clustering, but in some cases soft-clustering provides a better model in practice. Gaussian mixture model assumes a simple **linear superposition** of Gaussian components, aimed at providing a richer class of density models than the single Gaussian. Let's consider a single sample case and it can be expressed as follows:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu_k}, \boldsymbol{\Sigma}_k)$$

Let us introduce a K-dimensional binary random variable $\mathbf{z}$ having a 1-of-K representation in which a particular element z_k is equal to 1 and all other elements are 0. I will explain more about $\mathbf{z}$ later. It satisfied the following properties:

- $z_k \in \{0, 1\}$
- $\sum_k z_k = 1$

The marginal distribution over $\mathbf{z}$ is specified in terms of the mixing coefficients π_k, such that

$$p(z_k = 1) = \pi_k$$

, where the mixing coefficients must satisfy

$$0 \leq \pi_k \leq 1$$

and

$$\sum_{k=1}^{K} \pi_k = 1$$

in order to be valid probabilities. We can also write pdf of $\mathbf{z}$ in a product of mixing coefficient because it is a 1-of-K representaion.

$$p(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k} = \pi_k \because z_k \in \{0, 1\}$$

Similarly, the conditional distribution of $\mathbf{x}$ given a particular $\mathbf{z}$ can be modeled to be a Gaussian distribution.

$$p(\mathbf{x}|z_k = 1) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Also can be represented in the form

$$\begin{aligned} p(\mathbf{x}|\mathbf{z}) &= \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k} \\ &= \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \because z_k \in \{0, 1\} \end{aligned}$$

Finally, marginal data distribution can be obtrained by summing the joint distribution over all possible states of $\mathbf{z}$ to give

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) \\ &= \sum_{\mathbf{z}} p(\mathbf{z}) p(\mathbf{x}|\mathbf{z}) = \sum_{z_1, ..., z_K} p(z_1, ..., z_K) p(\mathbf{x}|z_1, ..., z_K) \\ &= \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \end{aligned}$$

Note that for every observed data point $\mathbf{x}_n$ there is a corresponding latent variable $\mathbf{z}_n$, which **indicates the membership of** $\mathbf{x}_n$. This can be represented as in Fig. 1.2.

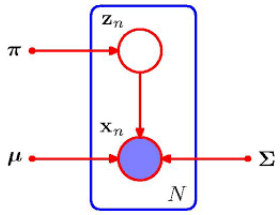

Figure 1.2: Graphical representation of GMM model. The GMM models a joint distribution $p(\mathbf{x}, \mathbf{z})$ in terms of a marginal distribution $p(\mathbf{z})$ and conditional distribution $p(\mathbf{x}|\mathbf{z})$ to model $p(\mathbf{x})$. Each $\mathbf{x}_n$ is coupled with $\mathbf{z}_n$

Now we can work with the joint distribution $p(\mathbf{x}, \mathbf{z})$ instead of the marginal distribution $p(\mathbf{x})$, which is hard to estimate directly as explained in §1.1.1.

Another quantity which plays a central role is the conditional proability of $\mathbf{z}$ given $\mathbf{x}$, $p(z_k = 1|\mathbf{x})$.

- $p(z_k = 1) = \pi_k$ can be viewed as a prior of $z_k = 1$
- $\gamma(z_k)$: assignment probability or responsibility. This quantity will be updated through the Bayes Theorem.

  $\rightarrow$ A simple explanation is that this is the classification result of $\mathbf{x}_n$.

$$\begin{aligned}\gamma(z_k) \equiv p(z_k = 1|\mathbf{x}) &\equiv \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\mathbf{x}|z_j = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}\end{aligned}$$

1.3.1 Maximum Likelihood

Suppose we have a data set of observations $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\}^T \in R^{N \times D}$ and we want to model the data distribution $p(\mathbf{X})$ using GMM. If we assume an i.i.d. data set, it can be expressed as follows:

$$p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

then its loglikelihood function is given by:

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

Before, maximizing the likelihood, it is worth to emphasize two issues in GMM: (i) *singularities* and (ii) *identifiability*.

Singularity Suppose that one of the components of the mixture model, let us say the j-th component, has its mean μ_j exactly equal to one of the data points so that $\mu_j = \mathbf{x}_n$ for some value of n. This data point will then contributes a term in the likelihood function of the form

$$\mathcal{N}(\mathbf{x}_n, \sigma_j^2 \mathbf{I}) = \frac{1}{(2\pi)^{1/2}} \frac{1}{\sigma_j}$$

If we consider the limit $\sigma_j \to 0$, then we see that this term goes to infinity and so the log likelihood function will also go to infinity. Thus the maximization of the log likelihood function will also go to infinity. Thus the maximization of the log likelihood function is not a well posed problem because such sigularities will walways be present and will occur whenever one of the Gaussian components collapses onto a specific data point.

Identifiability A further issue in finding MLE based solutions arises from the fact that for any given maximum likelihood solution, a K-component mixture will have a total ok $K!$ equivalent solutions corrsponding to the $K!$ ways of assigning K sets of parameters to K components. In other words, for any given point in the space of parameter values there will be a further $K! - 1$ additional points all of which give rise to exactly the same distribution.

1.3.2 Expectation Maximization for GMM

The goal of Expectation Maximization (EM) is to find maximum likelihood solutions for models having latent variables

- Suppose that it is hard to optimize $p(\mathbf{X}|\boldsymbol{\theta})$ directly.
- However, it is easier to optimize the complete-data likelihood function $p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$
- In this case, we can use **EM algorithm**. EM algorithm is a general technique for finding maximum likelihood solutions for latent variable models.

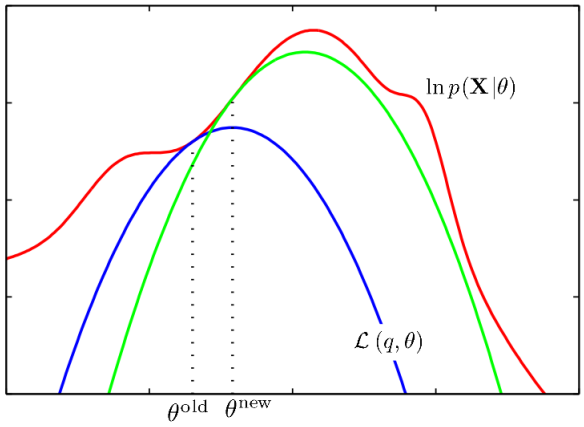

Figure 1.3: M-step of EM algorithm

Algorithm 1: EM algorithm for GMM

Initialize the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and mixing coefficients π_k and evaluate the initial value of the log likelihood.

for n **do**

E step: evaluate the responsibilities of $\mathbf{x}_n$ based on the current parameter values with the given parameters

$$\gamma(z_{nk}) = p(z_k = 1|\mathbf{x}_n) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

where z_{nk} denote the k-th component of $\mathbf{z}_n$

M step: maximize expectation

- $\boldsymbol{\mu}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n$
- $\boldsymbol{\Sigma}_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}})^T$
- $\pi_k^{\text{new}} = p(z_k = 1) = \frac{N_k}{N}$

Evaluate the log likelihood to check for convergence of parameters

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

1.4 Latent Variable Modeling

For each object x_i, we establish additional latent variable z_i which denotes the index of gaussian from which i-th object was generated. Then our model is

$$p(X, Z|\theta) = \prod_{i=1}^{n} p(x_i, z_i|\theta) = \prod_{i=1}^{n} p(x_i|z_i, \theta)p(z_i|\theta) = \prod_{i=1}^{n} \mathcal{N}(x_i|\mu_{z_i}, \sigma^2_{z_i})\pi_{z_i},$$

where $\pi_j = p(z_i = j)$ are prior probability of j-th gaussian and $\theta = \{\mu_j, \sigma_j, \pi_j\}_{j=1}^{K}$. If we know both X and Z then we can obtain explicit ML-solution:

$$\theta_{ML} = \underset{\theta}{\operatorname{argmax}}\, p(X, Z|\theta) = \underset{\theta}{\operatorname{argmax}} \log p(X, Z|\theta).$$

However, in practice, we don't know Z, but only know X. Thus, we need to maximize w.r.t. θ the log of incomplete likelihood

$$\log p(X|\theta) = \int q(Z) \log p(X|\theta) dZ \tag{1.1}$$

$$= \int q(Z) \log \frac{p(X, Z|\theta)}{p(Z|X, \theta)} dZ \tag{1.2}$$

$$= \int q(Z) \log \frac{q(Z)p(X, Z|\theta)}{q(Z)p(Z|X, \theta)} dZ \tag{1.3}$$

$$= \int q(Z) \log \frac{p(X, Z|\theta)}{q(Z)} dZ + \int q(Z) \log \frac{q(Z)}{p(Z|X, \theta)} dZ \tag{1.4}$$

$$= \underbrace{\int q(Z) \log \frac{p(X, Z|\theta)}{q(Z)} dZ}_{\text{ELBO, } \mathcal{L}(q,\theta)} + \mathrm{KL}(q(Z)||\log p(Z|X, \theta)) \tag{1.5}$$

The first line uses that Z is not related to X, so the integration does not affect X. Note that $p(X|\theta)$ in Eq.1.5 (1) does not depend on Z. Also note that p does not depend of q, **so maximizing ELBO is equal to minimizing the KL divergence**.

By using ELBO, we are able to maximize the incomplete likelihood. If you see the KL term, it is trying to minimize the divergence between $q(Z)$ and $p(Z)$ through maximizing ELBO.

1.4.1 ELBO

Note that $\mathrm{KL}(q(Z)||\log p(Z|X, \theta)) \geq 0$, thus $\mathcal{L}(q, \theta) \leq \log p(X|\theta)$. In other words, $\mathcal{L}(q, \theta)$ is a **lower bound** on $\log p(X|\theta)$.

Let's see ELBO in a different perspective

$$\begin{aligned}
\mathcal{L}(q, \theta) &= \int q(Z) \log \frac{p(X, Z|\theta)}{q(Z)} dZ \\
&= \int q(Z) \Big[\log p(X, Z|\theta) - \log q(Z) \Big] dZ \\
&= \int q(Z) \Big[\log p(Z|X, \theta) + \log p(X|\theta) - \log q(Z) \Big] dZ \\
&= \int q(Z) \log p(X|\theta) dZ + \int q(Z) \log \frac{p(Z|X, \theta)}{q(Z)} dZ \\
&= \log p(X|\theta) - \mathrm{KL}(q(Z)|p(Z|X, \theta))
\end{aligned}$$

To maximize the above equation, we need to minimize KL divergence.

1.4.2 Expectation Maximization

We want to maximize ELBO, $\mathcal{L}(q, \theta)$, to minimize KL divergence between $q(Z)$ and $\log p(Z|X, \theta)$.

$$\max_{q,\theta} \mathcal{L}(q, \theta) = \max_{q,\theta} \int q(Z) \log \frac{p(X, Z|\theta)}{q(Z)} dZ.$$

We start from initial point θ_0 and iteratively repeat (*i*) E-step and (*ii*) M-step, iteratively:

- E-Step: θ_0 is fixed.

$$q(Z) = \operatorname*{argmax}_q \mathcal{L}(q, \theta) = \operatorname*{argmin}_q \mathrm{KL}(q(Z)|p(Z|X, \theta)) = p(Z|X, \theta_0).$$

  - This is because, maximizing ELBO is equal to minimizing KL divergence and the minimum q can be achieved when q is equal to $p(Z|X, \theta_0)$.
  - Now, we just have to evaluate $p(Z|X, \theta_0)$.

- M-Step: q is fixed.

$$\theta_* = \operatorname*{argmax}_\theta \mathcal{L}(q, \theta) = \operatorname*{argmax}_\theta \mathbb{E}_{q(Z)}[\log p(X, Z|\theta)]$$

  - Can be accomplished by taking derivatives
  - Set $\theta_0 = \theta_*$ and go to the E-Step until convergence

1.4.3 Categorical Latent Variables

$z_i \in \{1, ..., K\}$

$$p(x_i|\theta) = \sum_{k=1}^{K} p(x_i|k, \theta) p(z_i = k|\theta)$$

is simply a finite mixture of distributions.

E-Step:

$$q(z_i = k) = p(z_i = k|x_i, \theta) = \frac{p(x_k|z_i = k, \theta) p(z_i = k|\theta)}{\sum_{l=1}^{K} p(x_i|z_i = l, \theta) p(z_i = l|\theta)}$$

M-Step:

$$\operatorname*{argmax}_\theta \mathbb{E}_{q(Z)}[\log p(X, Z|\theta)] = \sum_{i=1}^{n} \mathbb{E}_{q(z_i)}[\log p(x_i, z_i|\theta)] = \sum_{i=1}^{n} \sum_{k=1}^{K} q(z_i = k) \log p(x_i, k|\theta)$$

For GMM, we model $p(x|z)$ as Gaussian.

Chapter 2

Deep Generative Models

2.1 Variational Autoencoder

Our goal is to find the data distribution $p(X)$. Fig. 2.1 represents a general structure of deep generative model. As you can see, we first sample $z \sim p(z)$ and feed it into a deep neural network $f(z)$ and output x.

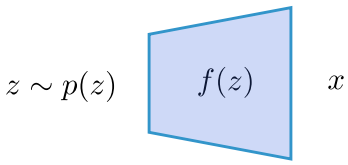

Figure 2.1: General structure of deep generative models. This model does not infer z from x.

VAE performs an inference by introducing a probabilistic encoder, called inference network. VAEs are generative model with a latent variable distributed according to some distribution $p(z_i)$. The observed variable is distributed according to a conditional distribution

$$p_\theta(x_i|z_i)$$

This conditioning means the latent variable values are the one most likely given the observations. We also create a distribution $q_\phi(z_i|x_i)$. We would like to be able to encode our data into the latent variable space. Let's model the distribution.

- $p_\theta(x_i|z_i) \sim \mathcal{N}(x_i|\mu(z_i), \sigma^2(z_i))$: A probabilistic decoder (or generative network, θ)
- $q_\phi(z_i|x_i)$: A probabilistic encoder (or inference network ϕ). We can choose a family of distributions for our conditional distribution q (*e.g.,* standard Gaussian distribution).

  $$q_\phi(z_i|x_i) = \mathcal{N}(z_i|\mu(x_i, W_1), \sigma^2(x_i, W_2)I),$$

  where W_1 and W_2 are network weights and collectively denoted as ϕ. We create a neural network to model the distribution q from our data in a non-linear manner. The outputs of the network are μ and σ.

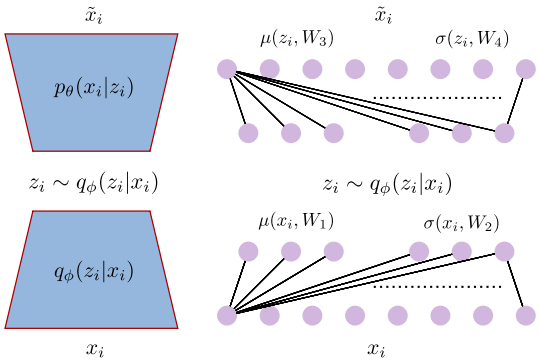

Figure 2.2: Overview of variational autoencoder.

$$
\begin{aligned}
p(X, Z|\theta) &= \prod_{i=1}^{n} \underbrace{p(x_i|z_i, \theta)}_{\text{Likelihood, Generator}} \underbrace{p(z_i|\theta)}_{\text{Prior on latent variable}} \\
&= \prod_{i=1}^{n} \mathcal{N}(x_i| \underbrace{\mu(z_i), \sigma^2(z_i)}_{\text{Non-linear}}) \mathcal{N}(z_i|0, I)
\end{aligned}
$$

Subsequently, marginal distributions can be expressed as follows under i.i.d. assumption:

$$
\begin{aligned}
p(X|\theta) &= \prod_{i=1}^{n} p(x_i|\theta) \\
&= \prod_{i=1}^{n} \int p(x_i, z_j|\theta) dz_j \\
&= \prod_{i=1}^{n} \int p(x_i|z_i, \theta) p(z_i|\theta) dz_i \\
&= \prod_{i=1}^{n} \int \mathcal{N}(x_i|\mu(z_i), \sigma^2(z_i)) \underbrace{\mathcal{N}(z_i|0, I)}_{\text{Mixture weight}} dz_i
\end{aligned}
$$

- As you can see, the marginal distribution $p(X|\theta)$ becomes a mixture of Gaussian (infinite mixture of Gaussian).
- Even though $p(x|z)$ and $p(z)$ are normal, $p(x)$ is not normal, because it is a mixture distribution.
- The non-linearity of Gaussian parameters (modeled by a neural network), conjugacy between the prior and the likelihood does not hold anymore.
- Again, μ and σ is non-linear function of z modeled by some non-linear neural network. The neural network works as a powerful non-linear parameter approximator (based on universal approximation theorem).
- Simple prior is used. Let's consider the data x is an image of 100×100 pixels. Then the covariance matrix has to be 10000×10000. Thus, it is common to set a simple prior such as the standard Gaussian (covariance matrix is diagonal matrix). However, even if we set a simple distribution, with the infinite mixture of Gaussian, we can model any distribution.

- VAE uses a global parametric model to predict the local variational parameters for each data point (amortized inference).
- It allows to convert complicated large-dimensional data distributions into simple lower-dimensional latent variable representations.

Bibliography