Fondamentaux de l'Apprentissage Automatique

Lecturer: Yann Chevaleyre — Lecture n°10-11 #
Scribe: Robin MICHARD (EM Algorith Include) Edgar FRIXONS (Auto-Encoder) — 07-08/12/2023

1 Introduction

Latent variable appears everywhere, but in this lecture, we will focus on generative problems (unsupervised learning).

Definition 1. *A generative problem is learning a distribution from datas* $(x_1, \dots, x_n) \in \mathbb{R}^d$ *in order to generate new points from this distribution.*

Definition 2. *A latent variable* z_i *is a missing information about an example* x_i*.*

Some example are :
— Making clustering without labels
— Dimension reduction as a projection from a larger dimension to a smaller
— Handling missing data
One of the best learning algorithm is **ExpectationMaximisation (EM)**

Definition 3. *Probabilistic Machine learning is when you assume that you data (x, y or both) was generated according to some generative model with unknown parameters.*

Probabilistic ML is the most appropriated method to handle latent variables. Learning is often done by Maximum Likelihood and Maximum A Posteriori Estimation.

Example Assume I have $f : [0,1] \to \mathbb{R}^2$ describing my curve.

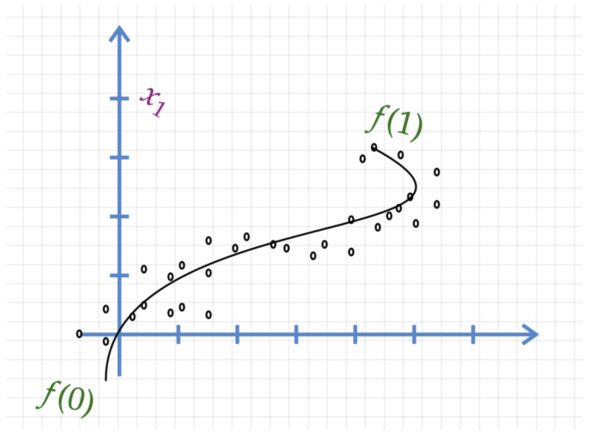


FIGURE 1 – Exemple dans $\mathbb{R}^2$

$$z_i \sim U_{ni}f([0,1])$$
$$\varepsilon_i \sim \mathcal{N}(0,I) \text{ with } \varepsilon_i \in \mathbb{R}^2$$
$$x_i = f(z_i) + \varepsilon_i$$
$$P(X \mid Z) = \mathcal{N}(X; f(X), I)$$
$$P(X) = \int p(X \mid Z)p(z)dz = \int_0^1 \mathcal{N}(X; f(z), I)dz$$
$$P(Z \mid X) = \frac{P(X \mid Z)}{P(Z)} \times P(X).$$

2 Gaussian mixtures and the EM algorithm

Definition 4. *The probability density function (PDF) of a multivariate normal distribution in p-dimensional space is given by the formula :*

$$f(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{p/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

Here :
— $\mathbf{x}$ *is the p-dimensional vector representing a random variable,*
— $(\mathbf{x}-\boldsymbol{\mu})^T$ *denotes the transpose of the difference vector,*
— $|\boldsymbol{\Sigma}|$ *is the determinant of the covariance matrix,*
— $\boldsymbol{\Sigma}^{-1}$ *is the inverse of the covariance matrix.*

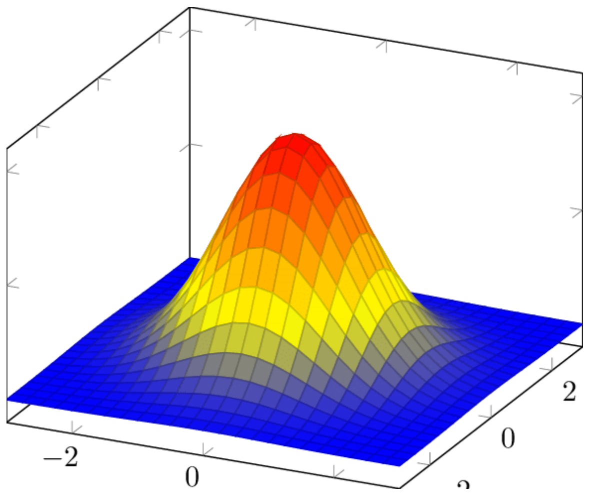


FIGURE 2 – Two dimensional gaussian

2.1 Gaussian Mixture Model

Definition 5. *The probability density function (PDF) of a Gaussian Mixture Model (GMM) is given by the formula :*

$$f(\mathbf{x}; \boldsymbol{\Theta}) = \sum_{k=1}^{K} \pi_k \cdot \frac{1}{(2\pi)^{D/2} |\boldsymbol{\Sigma}_k|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right)$$

where :
— $\mathbf{x}$ *is the observation vector,*
— D *is the dimensionality of the data,*
— $\mathbf{\Theta}$ *represents the set of all parameters* $\{\boldsymbol{\mu}_k, \mathbf{\Sigma}_k, \pi_k\}$ *for* $k = 1, 2, \ldots, K$.

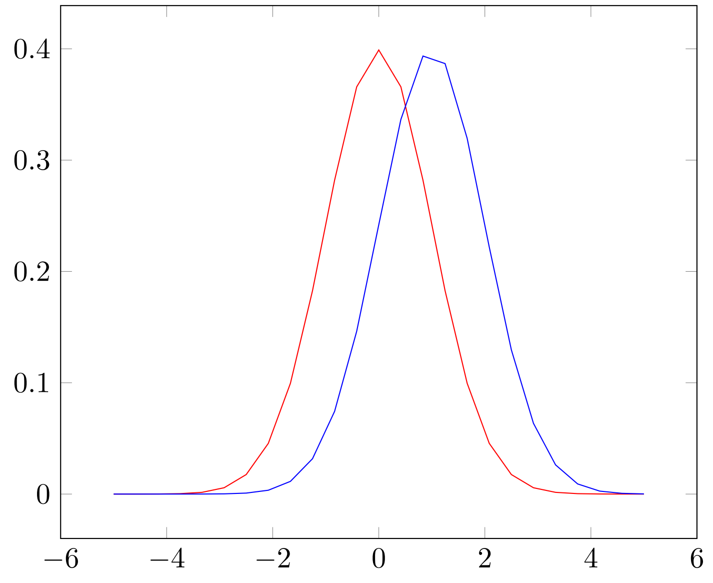
Remark : GGMM is the same as **LDA** but without the class information so it's unsupervised.

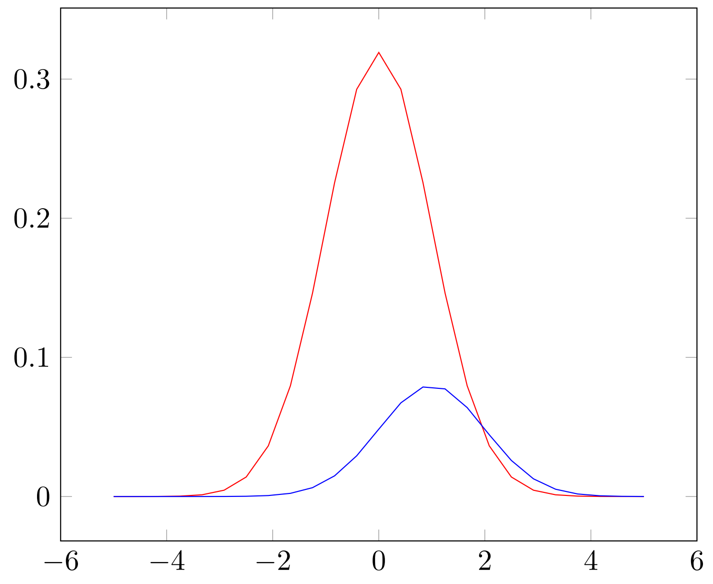
Exercise

$K = 2$ gaussian	$d = 1 \quad (x_1 \cdots x_N) \in \mathbb{R}$
$\mu_1 = 0$	$\mu_2 = 1$
$\sigma_1 = 1$	$\sigma_2 = 1$
$\pi = p\left(Z_i = 1\right) = 80\%$	$P\left(Z_i = 2\right) = 1 - \pi$

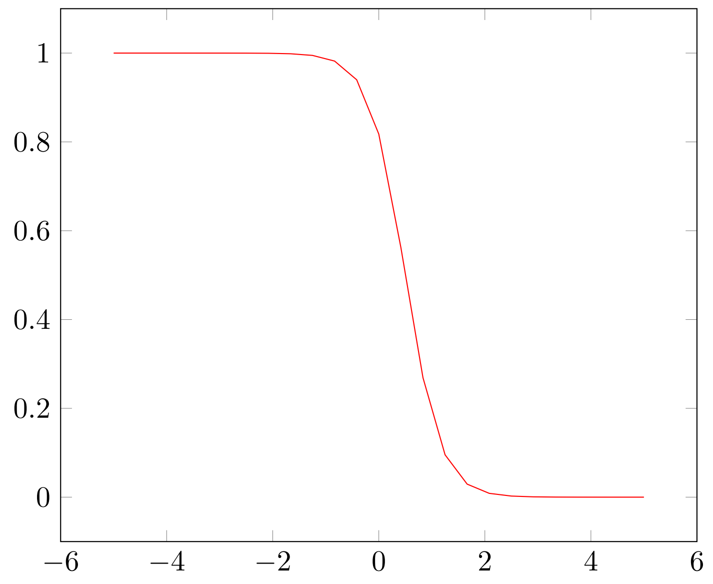
1. Draw informally $P(x \mid z = 1), P(x \mid z = 2)$ on a graph.
2. Draw $P(x, z = 1)$ and $P(x, z = 2)$ on another graph.
3. Draw $P(Z = 1 \mid X)$
   Hint : $P(Z = 1 \mid X) = \frac{P(X, Z=1)}{P(x)} = \frac{P(X|Z=1)P(Z=1)}{\pi P(X|Z=1) + (1-\pi)P(X|Z=2)}$
4. Compute and simplify $P(Z = 1 \mid X)$

Solutions

1. 



2.



3.

4. $P(z=1 \mid x) = \frac{P(x|z=1)\pi}{\pi \cdot P(x|z=1)+(1-\pi)P(x|z=2)}$

$= \frac{\pi \exp\left(-\frac{1}{2}(x-\mu_1)^2\right)}{\pi \exp\left(-\frac{1}{2}(x-\mu_1)^2\right)+(1-\pi)\exp\left(-\frac{1}{2}(x-\mu_2)^2\right)}$

$= \frac{1}{1+\frac{1-\pi}{\pi}\exp\left\{-\frac{1}{2}(x-\mu_2)^2+\frac{1}{2}(x-\mu_1)^2\right\}}$

$= \frac{1}{1+exp\left(x(\mu_2-\mu_1)+\frac{\mu_2^2-\mu 1^2}{2}+ln\frac{1-\pi}{\pi}\right)}$

$= \frac{1}{1+exp(ax+b}$

With $a = \mu_2 - \mu_1$ and $b = \frac{\mu_2^2-\mu_1^2}{2} + ln\frac{1-\pi}{\pi}$

Definition 6. *The complete log-likelihood of a Gaussian Mixture Model (GMM) is given by :*

$$\mathcal{L}(q, \mathbf{\Theta}; X) = \sum_{i=1}^{N} \log\left(\sum_{k=1}^{K} q_i(k) \cdot \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \mathbf{\Sigma}_k)\right)$$

Here, $\mathbf{x}_i$ *is a data point,* $\mathbf{\Theta} = \{\boldsymbol{\mu}_1, \mathbf{\Sigma}_1, \ldots, \boldsymbol{\mu}_K, \mathbf{\Sigma}_K\}$ *are the GMM parameters, and*

$\mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the probability density function of the multivariate normal distribution.

We have that $q = (q_1, \ldots, q_N)$ where $q_i(k) = P_{\boldsymbol{\Theta}}(z_i = k \mid x_i)$

The CMM assumes that we know $\{z_1, z_2, \ldots, z_N\}$ so we can't compute it in our case. So we will estimate $p(z_i = k \mid x_i; \hat{\boldsymbol{\Theta}})$ we will estimate it with $q_i(k)$

Definition 7. *The Eexpected CLL is given by the formula :*

$$ECLL(q, \boldsymbol{\Theta}; X, Z) = \sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \cdot (\log(q_i(k)) + \log(\mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)))$$

Here :
- *$Z = \{z_1, z_2, \ldots, z_N\}$ is the set of latent variables,*
- *z_{ik} is the i-th element of the one-hot encoded vector z_i indicating the assignment of the i-th data point to the k-th component*

Remark : $\mathcal{L}(\boldsymbol{\Theta}; X)$ is convex in $(\mu_1, \ldots, \mu_N)$

2.2 The EM Algorithm

The Expectation-Maximization (EM) algorithm is an iterative optimization algorithm for finding maximum likelihood estimates of parameters in models with latent variables. It consists of two main steps : the E-step (Expectation step) and the M-step (Maximization step).

Initialization

1. Initialize the parameters of the model, denoted as $\boldsymbol{\Theta}^{(0)}$.

Iteration (for $t = 1, 2, \ldots$)

1. **E-step (Expectation step) :** for each $i \in 1 \ldots N, k \in 1 \ldots K$ compute $q_i(k) = p(z_i = k \mid x_i; \boldsymbol{\Theta}^{(t)})$
2. **M-step (Maximization step) :** Update the parameters to maximize the expected complete log-likelihood :

$$\boldsymbol{\Theta}^{(t)} = \arg\max_{\boldsymbol{\Theta}} \text{ECLL}(q, \boldsymbol{\Theta}; X, Z)$$

Convergence

Repeat the E-step and M-step until the change in the log-likelihood or parameter values falls below a specified threshold, or until a maximum number of iterations is reached.

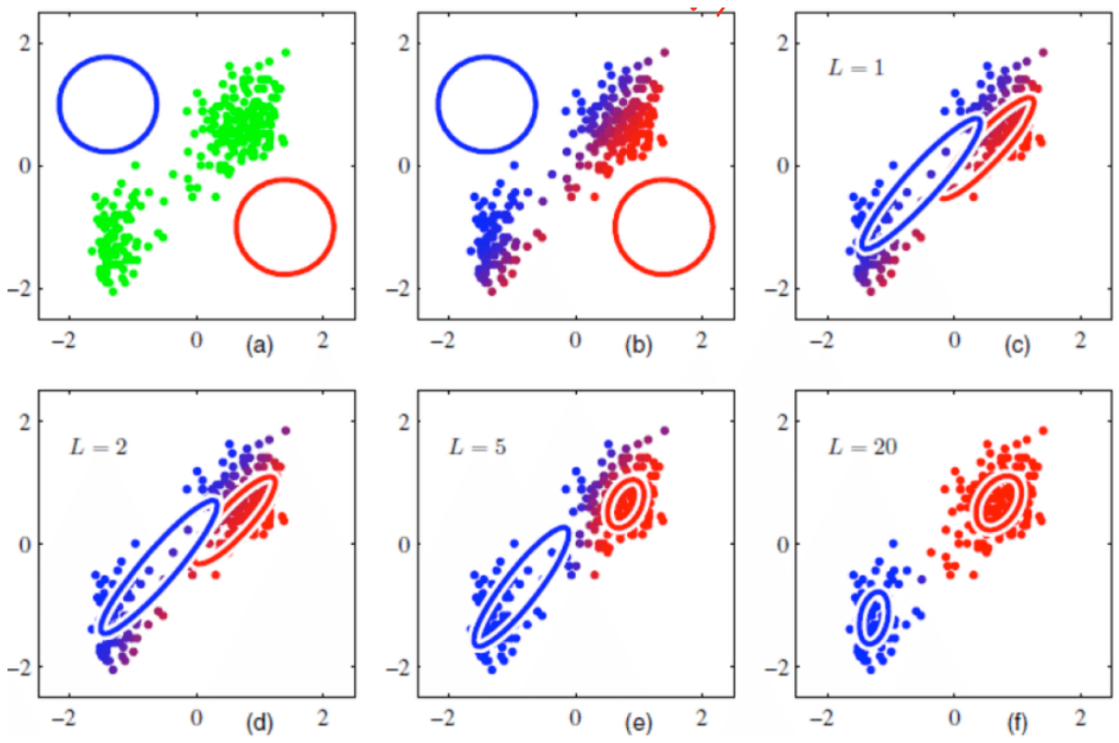


FIGURE 3 – Illustration EM Algorithm with $K = 2$

2.3 The ELBO

We really want to optimise the likelihood $P_{\mathbf{\Theta}}(x_1, \ldots, x_N)$ but we saw that instead we optimise the expected CLL. Why is that ?

Recall that $\mathcal{L}(\mathbf{\Theta}; X) = \sum_{i=1}^{N} \mathcal{L}(\mathbf{\Theta}; X, i)$ where $\mathcal{L}(\mathbf{\Theta}; X, i) = E_{z_i \sim \pi_i}[P(X = x_i, z_i = k \mid \mathbf{\Theta})]$

Let's pick a single point x and compute its LL. Lets write $q_x(k) \sim P_{\mathbf{\Theta}}(z = k \mid X = x)$

$$logP_{\mathbf{\Theta}}(x) = ln \sum_{k=1}^{K} P_{\mathbf{\Theta}}(x, z = k) = ln \sum_{k=1}^{K} q_x(k) \times \frac{P_{\mathbf{\Theta}}(x, z = k)}{q_x(k)}$$

Jensen's inequality give us

$$lnE_{k \sim q_k} \left[\frac{P_{\mathbf{\Theta}}(x, z = k)}{q_x(k)} \right] \geq E_{k \sim q_k} ln \left[\frac{P_{\mathbf{\Theta}}(x, z = k)}{q_x(k)} \right]$$

$$logP_{\mathbf{\Theta}}(x) \geq \underbrace{\underbrace{\sum_{k=1}^{K} q_x(k) lnP_{\mathbf{\Theta}}(x, z = k)}_{ECLL} - \underbrace{\sum_{k=1}^{K} q_x(k) lnq_x(k)}_{Entropy}}_{\text{ELBO}(x, \mathbf{\Theta}, q_x)}$$

We can now measure how close the ELBO and the LL are.

$$\begin{aligned}
lnP_{\mathbf{\Theta}}(x) - \text{ELBO}(x, \mathbf{\Theta}, q_x) &= lnP_{\mathbf{\Theta}}(x) - E_{k\sim q_k} ln\left[\frac{P_{\mathbf{\Theta}}(x, z=k)}{q_x(k)}\right] \\
&= E_{k\sim q_k} ln\left[\frac{P_{\mathbf{\Theta}}(x)q_x(k)}{P_{\mathbf{\Theta}}(x, z=k)}\right] \\
&= E\left[ln\frac{q_k(k}{P_{\mathbf{\Theta}}(z=k \mid x)}\right] \\
&= \text{KL}(q_x(z)||P_{\mathbf{\Theta}}(z=k \mid x))
\end{aligned}$$

Where KL is the Kullback-Leibler divergence.
Knowing that $KL(q||q') = 0 \Leftrightarrow q = q'$
So we have

$$\begin{aligned}
&lnP_{\mathbf{\Theta}}(x) = \text{ELBO}(x, \mathbf{\Theta}, q_x) \Leftrightarrow q_x(z) = P_{\mathbf{\Theta}}(z=k \mid x) \\
&\Rightarrow \arg\max_{q_x} \text{ELBO}(x, \mathbf{\Theta}, q_x) = P_{\mathbf{\Theta}}(z=k \mid x) \\
&\Rightarrow \arg\max_{\mathbf{\Theta}} \text{ELBO}(x, \mathbf{\Theta}, P_{\mathbf{\Theta}}(z=k \mid x)) = \arg\max_{\mathbf{\Theta}} lnP_{\mathbf{\Theta}}(x) \\
&\Rightarrow \arg\max_{\mathbf{\Theta}, q_x} \text{ELBO}(x, \mathbf{\Theta}, q_x) = \arg\max_{\mathbf{\Theta}} lnP_{\mathbf{\Theta}}(x)
\end{aligned}$$

So the EM algorithm become :

Initialization

1. Initialize the parameters of the model, denoted as $\mathbf{\Theta}^{(0)}$.

Iteration (for $t = 1, 2, \dots$)

1. **E-step (Expectation step) :** for each $i \in 1 \dots N, k \in 1 \dots K$ compute $q_i(k)$ with $q_1, \dots, q_N = \arg\max_{q_1,\dots,q_N} \sum_i \text{ELBO}(x_i, \mathbf{\Theta}^{(t)}, q_i)$
2. **M-step (Maximization step) :** Update the parameters to maximize the expected complete log-likelihood :

$$\mathbf{\Theta}^{(t)} = \arg\max_{\mathbf{\Theta}} \sum_i \text{ELBO}(x_i, \mathbf{\Theta}^{(t)}, q_i)$$

Remarks :
— Because $\text{ELBO}(x_i, \mathbf{\Theta}^{(t)}, q_i) = lnP_{\mathbf{\Theta}}(x_i) - \text{KL}(q_i(z), P_{\mathbf{\Theta}}(z \mid x_i)$ the E-step minimizes the KL and the M-step maximizes tje LL while possibly increasing the KL.
— Each step (E and M) increases the ELBO. Unless we get stuck in a local minimum, this will reach the opotimal ELBO which coincide with the optimal LL.

3 Variational Auto Encoder and Diffusion Model

3.1 Auto-Encoders and Variational Auto Encoders (VAE)

Definition 8. ***Auto Encoder*** *: Model with an encoder and a decoder. The standard encoder learns to reduce the dimensionality of the examples of a dataset. It works according to this scheme :*

$$x \to Enc_\phi(x) \to z \to Dec_\psi(x) \to \hat{x}$$

With :

— x *the initial image.*

— Enc_ϕ *the encoder with the parameters* ϕ*.*

— z *the latent image.*

— Dec_ψ *the decoder with the parameters* ψ*.*

— $\hat{x}$ *the exit image.*

From this definition, we can train the model in the following manner :

$$\arg\min_{\phi,\psi} \sum_{i=1}^{N} \|x_i - \hat{x}_i\|^2$$

However, for this encoder, the latent distribution can be very weird, it can have any kind of shape and defining function.

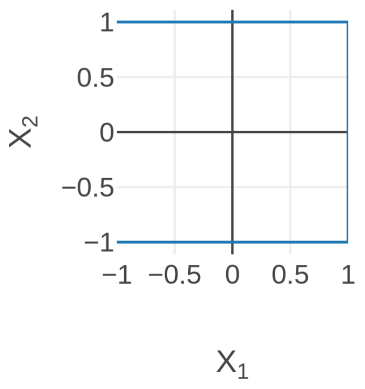


FIGURE 4 – Example of function the latent distribution can follow

However, this limits the possibility for use, thus we can want to change the latent distribution to have something more usable.

Definition 9. ***Variational Auto Encoder (VAE)*** *: A VAE is an Auto Encoder with a general shape for its latent space. This shape, or this distribution, is similar to a Gaussian Noise.*

From a mathematical point of view :

$$p_\theta(x|z) = \mathcal{N}(x|\mu_i, I) = \mathcal{N}(x|Dec_\theta(z), I)$$
$$x = Dec_\theta(z) + \epsilon \text{ with, } \epsilon \sim \mathcal{N}(0, I)$$

3.2 The training of the model

3.2.1 Training the Decoder alone

Assume we have a dataset $\{(x_i, z_i)\}_{i=1...N}$ with $x_i \in \mathbb{R}^d$ and $z_i \in \mathbb{R}^k$, $k \leq d$.
We search $\hat{\theta}$ such that :

$$\begin{aligned}
\hat{\theta} &= \arg\max_{\theta} \sum_{i=1}^{N} log p_{\theta}(x_i|z_i) \\
&= \arg\max_{\theta} \sum_{i=1}^{N} log \mathcal{N}(x_i|Dec_{\theta}(z_i), I) \\
&= \arg\max_{\theta} \sum_{i=1}^{N} -\frac{1}{2}\|x_i - Dec_{\theta}(z_i)\|^2 \\
\hat{\theta} &= \arg\min_{\theta} \sum_{i=1}^{N} \|x_i - Dec_{\theta}(z_i)\|^2
\end{aligned}$$

We assumed to have access to the z_i, but these are of course extracted from the trained encoder.

3.2.2 Training the model with ELBO

First, we look at the decoding function and the decoding probability : $p_{\theta}(x|z)$.
There we assume to have $z_i \sim \mathcal{N}(O, I)$.
We have : $p_{\theta}(x) = \int_{\mathbb{R}^k} p_{\theta}(x|z)p(z)dz$.
However, this integral is hard because $p_{\theta}(x|z)$ is close to zero for most of the examples we can take. Thus, the direct training to find the previously talked about $\hat{\theta}$ is complicated.

Hence the training is made with ELBO, which requires $q(z|x)$, which is the encoder probability. This will translate in our problem as the following equation :

$$q_{\phi}(z|x) = (N)(z|\mu_{\psi}(x), \sigma^2_{\phi}(x))$$

Then the ELBO objective is :

$$\begin{aligned}
log p_{\theta}(x) &\geq ELBO(x, \theta, q_{\phi}) \\
&= \mathbb{E}_{z\sim q_{\phi}(|z)}\left[log \frac{p_{\theta}(x,z)}{q_{\phi}(z|x)}\right] \\
&= \mathbb{E}_{z\sim q_{\phi}(|z)}\left[log(p_{\theta}(x,z)) - \frac{q_{\phi}(z|x)}{p(z)}\right] \\
&= \mathbb{E}_{z}\left[log(p_{\theta}(x,z))\right] - KL(q_{\phi}(z|x), p(z)) \\
&= \mathbb{E}_{z\sim q_{\phi}(|z)}\left[-\frac{1}{2}\|x - Dec_{\theta}(s)\|^2\right] - \frac{1}{2}\|\mu_{\theta}(x)\|^2 + tr(\Sigma_{\phi}(x)) - k - log(|\Sigma_{\phi}(z)|) + cst
\end{aligned}$$

However, we have a problem now : we will not backproagate because on ϕ because we follow the law given by ϕ to get our examples. For this we use the **reparameterization trick** :

Definition 10. ***Reparameterization Trick*** *: This is a trick we can use when we have troubles with ELBO because we use the parameters to get our examples. It is described by two things :*

— *Problem : The ELBO algorithm cannot backpropagate because the parameters we want to change only appear in the distribution and so in the creation of our examples.*

— *Solution :* $z \sim q_\phi(|x) = \mathcal{N}(\mu_\psi(x), \sigma^2_\psi(x))$ *is identically distributed to* $z' = \mu_\psi(x) + \sigma^2_\psi(x)\epsilon$ *with* $\epsilon \sim \mathcal{N}(0, I)$.

In the end, we have the following algorithm for training :

Algorithm 1 VAE_Training(x, $iters$)

```
Init : $\hat{\psi}$ and $\hat{\theta}$
for $i \leq iters$ do
    $\psi \leftarrow \arg\max_{\hat{\phi}} ELBO(x, \hat{\psi}, \theta)$
    $\theta \leftarrow \arg\max_{\hat{\theta}} ELBO(x, \psi, \hat{\theta})$
    $i \leftarrow i + 1$
end for
```

3.3 Denoising Diffusion Models

Definition 11. ***Denoising Diffusion Model*** *: It is a model with two main processes :*

— *A fixed forward diffusion process, noted q, that will add noise to the layer it is applied to.*

— *A generative reverse denoising process, noted p, that will denoise the layer it is applied to.*

For more precision, we have : $q(x_0)$ = distribution of the images of the dataset

$$\begin{aligned}
&q(x_t|x_{t-1}) = \mathcal{N}(x_t|\sqrt{1-\beta_t}x_{t-1}, \beta_t I) \\
&x_t = x_{t-1} \times \sqrt{1-\beta_t} + \beta_t \times \epsilon_t \\
&q(x_1, ..., x_T|x_0) = \prod_{i=1}^{T} q(x_t|x_{t-1}) \\
&\text{with : } \epsilon_t \sim \mathcal{N}(0, I), 0 \leq \beta_t \leq 1 \text{ and } \beta_{t-1} \leq \beta_t
\end{aligned}$$

Also, for having lighter equations, we define :

$$\begin{aligned}
&\alpha_t = 1 - \beta_t \\
&\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i \\
&x_t = \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\epsilon_t = \sqrt{\bar{\alpha}_t} \times x_0 + \sqrt{1-\bar{\alpha}_t} \times \epsilon \\
&q(x_t|x_0) = \mathcal{N}(x_t|\sqrt{\bar{\alpha}_t}x_0, 1 - \bar{\alpha}_t I)
\end{aligned}$$

With it, we have two algorithms :

— One for training the VAE :

— One for sampling a new image :

Algorithm 2 training($q(x_0)$, $iters$)

```
for $i \leq iters$ do
    $x_0 \sim q(x_0)$
    $t \sim Uniform(\{1, \dots, T\})$
    $\epsilon \sim \mathcal{N}(O, I)$
    Take gradient step on :
        $\nabla_\theta \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, t)\|^2$
    $i \leftarrow i + 1$
end for
```

Algorithm 3 sampling()

```
$x_T \sim \mathcal{N}(0, I)$
for $t = T, \dots, 1$ do do
    $z \sim \mathcal{N}(0, I)$ if $t > 1$, else $z = 0$
    $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) + \sigma_t z$
end for
return $x_0$
```

Démonstration. We would want to compute q to have the best p, however this approach is too hard to be realistic. Instead, we reduce the problem to $q(x_{t-1}|x_t, x_0)$.
Thanks to the Conditional Bayes Rule :

$$\begin{aligned}
q(x_{t-1}|x_t, x_0) &= q(x_t|x_{t-1}, x_0)\frac{q(x_{t-1}|x_0)}{q(x_t, x_0)} \qquad = \mathcal{N}(x_{t-1}|\hat{\mu}_t(x_t, x_0), \hat{\beta}_t I) \\
\hat{\mu}_t(x_t, x_0) &= \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t} x_0 \\
x_0 &= \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1 - \bar{\alpha}_t}\epsilon_t) \\
\Rightarrow \hat{\mu}_t &= \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}}\epsilon_t)
\end{aligned}$$

At the end we want $q(x_{t-1}, x_t)$ to be close to $p_\theta(x_{t-1}, x_t)$. We want to maximize the

ELBO, or minimize -ELBO.

$$
\begin{aligned}
-ELBO &= \mathbb{E}_q\left[-log\frac{p_\theta(x_0,...,x_t}{q(x_1,...,x_t|x_0)}\right] \\
&= \mathbb{E}_q\left[-logp_\theta(x_T) - \sum_{t\geq 1} log\frac{p_\theta(x_{t-1}|x_t}{q(x_t|x_{t-1})}\right] \\
&= \mathbb{E}_q\left[-logp_\theta(x_T) - \sum_{t\geq 2} log\frac{p_\theta(x_{t-1}|x_t}{q(x_t|x_{t-1},x_0)} - log\frac{p_\theta(x_0,x_1)}{q(x_1,x_0)}\right] \\
&= \mathbb{E}_q\left[-logp_\theta(x_T) - \sum_{t\geq 2} log\frac{p_\theta(x_{t-1}|x_t}{q(x_{t-1}|x_t,x_0)}\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)} - log\frac{p_\theta(x_0,x_1)}{q(x_1,x_0)}\right] \\
&= \mathbb{E}_q\left[-logp_\theta(x_T) - \sum_{t\geq 2} log\frac{p_\theta(x_{t-1}|x_t}{q(x_{t-1}|x_t,x_0)} - logp_\theta(x_0,x_1)\right] \\
&= \mathbb{E}\left[KL(q(x_T|x_0),p(x_T)) + \sum_{t\geq 2} KL(q(xt-1|x_t,x_0),p_\theta(x_t-1|x_t)) - logp_\theta(x_0,x_1)\right] \\
&= \mathbb{E}\left[L_T + \sum_{t\geq 2} L_{t-1} - L_0\right]
\end{aligned}
$$

Let us focus on L_{t-1} :

$$
\begin{aligned}
L_{t-1} = KL(q(xt-1|x_t,x_0),p_\theta(x_t-1|x_t)) \\
= \|\mu_t(x_t,x_0) - \tilde{\mu_0}(x_t)\|^2 \times cst \\
= \|\epsilon_t - \epsilon_\theta(x_t,t)\|^2 \times cst \\
= \|\epsilon_t - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_0,t)\|^2
\end{aligned}
$$

□