Variational Autoencoder Mathematics

Andreas Spanopoulos
Demetrios Konstantinidis

andrewspanopoulos@gmail.com
demetris.konst@gmail.com

December 2, 2020

Introduction

The **Variational Autoencoder** (aka **VAE**) is a generative model. This means that it is a model which produces new unseen data. Unlike the normal Autoencoder, VAE focuses on understanding the distribution of a smaller representation of the data. This lower-dimensional representation of the data is known as “latent vector **z**”.

The dimension of the latent vector z is a hyperparameter which we choose along with the architecture of the Network. Keep in mind that we don’t want **z** to be too large. It should be a relatively small vector, so that an information bottleneck is created. One other reason for **z** being small, is that we want to be able to sample easily new vectors, without having to take into consideration many features.

With that said, the question arises: How can we pick the values of **z** which will make sense, that is, which will generate a new data point from the distribution of our original data?

Here is the beauty of the **Variational Autoencoder**: We will learn the distribution of **z**. That is, for every component of **z**, we will learn a mean and a standard deviation.

Suppose **z** has k components:

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_k \end{bmatrix}$$

Then, the mean and standard deviation vectors are defined as:

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_k \end{bmatrix}, \quad \sigma = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_k \end{bmatrix}$$

Our goal is to learn the μ and σ vectors in order to be able to sample **z** as follows

$$z = \mu + \epsilon \odot \sigma$$

where $\epsilon \sim N(0, 1)$ is a gaussian with mean 0 and standard deviation 1.

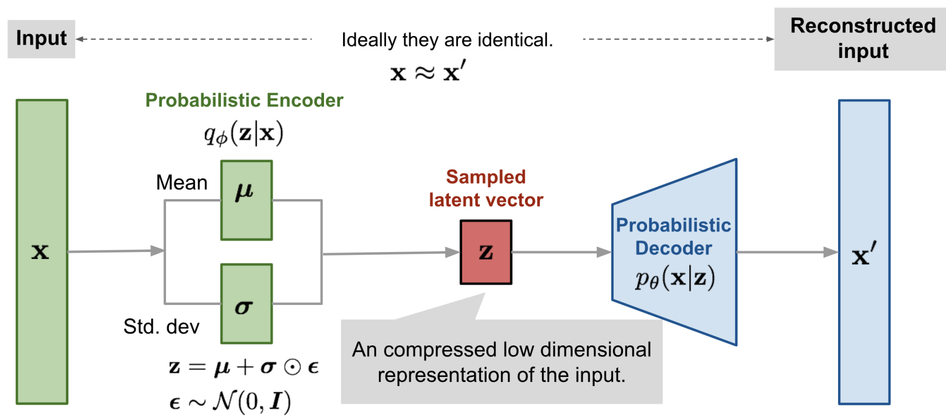

Figure 1: This picture demonstrates the architecture of a Variational Autoencoder. The input $\mathbf{x}$ gets fed in a Probabilistic Encoder $q_\phi(z|x)$, which in turns connects with the μ and σ layers. Note that usually there is a encoder Network before the mean and std layers, but here in the figure it is ommitted. Then, they sample $\mathbf{z}$ which in turn is fed to the Probabilistic Decoder $p_\theta(x|z)$. The result is then fed to an output layer which represents the reconstructed input data. The original picture can be found here.

Brief Explanation of architecture

The architecture of a **VAE** is briefly portrayed in Figure 1. Let's take a closer look in each part:

1. The encoder part consists of a Probabilistic Encoder $q_\phi(z|x)$. Given some parameters ϕ (which are parameters of the model), $q_\phi(z|x)$ models the probability of obtaining the latent vector $\mathbf{z}$ given input data $\mathbf{x}$. Afterwards, it connects to the μ and σ layers, as there might a whole encoder network before those.

2. The latent vector $\mathbf{z}$.

3. The decoder part which consists of a Probabilistic Decoder $p_\theta(x|z)$. As with the probabilistic encoder, given some parameters θ which are parameters of the model, we want to learn the probability of obtaining a data point $\mathbf{x}$ given a latent vector $\mathbf{z}$.

4. The reconstructed input $\hat{x}$.

Loss function

The loss function of the VAE is:

$$L(\theta, \phi, x) = -E_{z \sim Q_\phi(z|x)} [\log(P(x|z))] + D_{KL} [Q_\phi(z|x) \parallel P(z)]$$

It may seem daunting at first, but if we break it down into pieces then it gets much simpler.

KL-Divergence and multivariate Normal Distribution

Let's start by explaining what the second term of the loss function is. The **K**ullback **L**eiber Divergence, also known as Relative Entropy, is a measure of similarity between two probability distributions. It is denoted by $D_{KL}(\cdot \parallel \cdot)$, its unit of measure it called **nat** and it can computed by the formula (for discrete probability distributions):

$$D_{KL}(P \parallel Q) = \sum_x P(x) \log\left(\frac{P(x)}{Q(x)}\right) \tag{1}$$

Of course, this implies that $D_{KL}(P \parallel Q) \neq D_{KL}(Q \parallel P)$.

Now, let's suppose that both P, Q are multivariate normal distributions with means μ_1, μ_2 and covariance matrices Σ_1, Σ_2:

$$\begin{aligned}
P(x) &= N(x;\, \mu_1,\, \Sigma_1) = \frac{1}{\sqrt{(2\pi)^k |\Sigma_1|}} e^{-\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)} \\
Q(x) &= N(x;\, \mu_2,\, \Sigma_2) = \frac{1}{\sqrt{(2\pi)^k |\Sigma_2|}} e^{-\frac{1}{2}(x-\mu_2)^T \Sigma_2^{-1}(x-\mu_2)}
\end{aligned}$$

where k is the magnitude (length) of vector x.
Hence

$$\begin{aligned}
\log(P(x)) &= \log\left(\frac{1}{\sqrt{(2\pi)^k |\Sigma_1|}} e^{-\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)}\right) \\
&= \log\left(\frac{1}{\sqrt{(2\pi)^k |\Sigma_1|}}\right) + \log\left(e^{-\frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)}\right) \\
&= -\frac{k}{2}\log(2\pi) - \frac{1}{2}\log(|\Sigma_1|) - \frac{1}{2}(x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)
\end{aligned}$$

Following the exact same steps, we also get that

$$\log(Q(x)) = -\frac{k}{2}\log(2\pi) - \frac{1}{2}\log(|\Sigma_2|) - \frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)$$

With the help of the above equalities, expanding (1) yields:

$$\begin{aligned} D_{KL}(P\|Q) &= \sum_x P(x)\left[\log(P(x)) - \log(Q(x))\right] \\ &= \sum_x P(x)\left[\frac{1}{2}\log\left(\frac{|\Sigma_2|}{|\Sigma_1|}\right) - \frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right. \\ &\qquad\left. + \frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right] \end{aligned}$$

We can rewrite the above term an an Expectation over P:

$$\begin{aligned} D_{KL}(P\|Q) = E_P&\left[\frac{1}{2}\log\left(\frac{|\Sigma_2|}{|\Sigma_1|}\right) - \frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right. \\ &\qquad\left. + \frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right] \end{aligned}$$

Since the logarithmic term is independent of x, we can move it outside the expectation. This leaves us with

$$\begin{aligned} D_{KL}(P\|Q) = &\ \frac{1}{2}\log\left(\frac{|\Sigma_2|}{|\Sigma_1|}\right) \\ &-\frac{1}{2}E_P\left[(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right] \\ &+\frac{1}{2}E_P\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right] \end{aligned}$$

Let's now try to simplify the 2nd and 3rd terms of the above expression.

First, we have to recall the trace function and some of its properties. The trace of a square matrix A, denoted as $tr(A)$, is the sum of the elements along the main diagonal of A. The properties of the trace function which we will need are:

1. Trace of scalar: Considering the scalar as a 1×1 matrix, gives: $x = tr(x)$
2. Trace of Expectation: From 1: $E[x] = E[tr(x)] \Rightarrow tr(E[x]) = E[tr(x)]$
3. Cyclic Property: $tr(ABC) = tr(CAB)$

Having these properties in mind, we are now ready to simplify the expectation terms computed before during the simplification of the KL Divergence.

- Term 2. Note that the matrix multiplications inside the expectations reduce to a scalar value.

$$\begin{aligned} E_P\left[(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right] &\stackrel{(1)}{=} E_P\left[tr\left((x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right)\right] \\ &\stackrel{(3)}{=} E_P\left[tr\left((x-\mu_1)(x-\mu_1)^T\Sigma_1^{-1}\right)\right] \\ &\stackrel{(2)}{=} tr\left(E_P\left[(x-\mu_1)(x-\mu_1)^T\Sigma_1^{-1}\right]\right) \end{aligned}$$

  Σ_1^{-1} is independent from the expectation over P, so it can be moved outside, giving:

$$E_P\left[(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right] = tr\left(E_P\left[(x-\mu_1)(x-\mu_1)^T\right]\Sigma_1^{-1}\right)$$

  But the term $E_P\left[(x-\mu_1)(x-\mu_1)^T\right]$ is equal to the Covariance Matrix Σ_1, thus yielding

$$E_P\left[(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right] = tr\left(\Sigma_1\Sigma_1^{-1}\right) = tr(I_k) = k$$

- Term 3. Again, note that the matrix multiplications inside the expectations reduce to a scalar value.