Rahul Vernwal Follow
Jun 4, 2018 · 4 min read

An Introduction To Variational Auto-Encoder (VAE)

So, it's been a month now, and I badly feel the need to document my stuff. Chances are that I will forget everything, maybe in a month or two after moving from here. I preferred writing a story to make it somewhat less boring.

Just for the introduction, I started my internship at Personal Robotics Lab, University of Washington on 1st May '18. I am yet not sure of the exact problem formulation, it's all about the experiments and results. My only aim is to create some productive model out of this internship which could be used in future, maybe by other researchers interested in the same field.

Initially, I was investigating the existing CVAE model (Link) in various environments and parallely creating packages for 7D problems (basically arising from HERB's arm, a robot in our lab). With a very vague idea of what CVAE does, I was experimenting stuff on the model. I became so familiar with it, that I could even tell you the dimension of each and every layer there. So, in brief the paper proposes a model to sample points in promising regions given the condition vectors and your start, goal configuration.

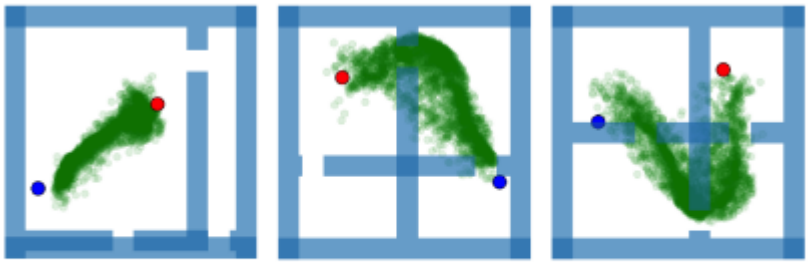
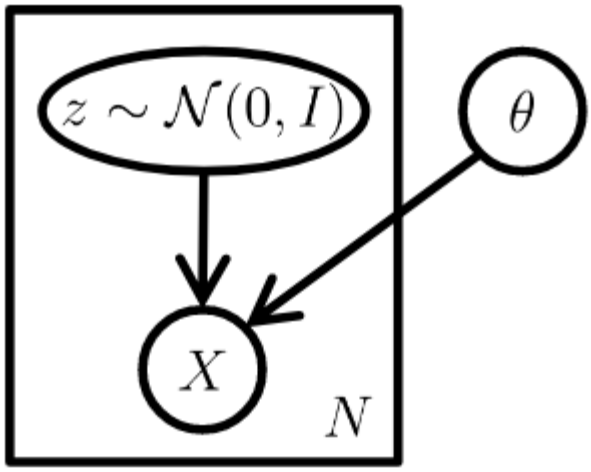


Fig. 7: Example learned distributions for the narrow passage problem, conditioned on the initial state (red), goal state (blue), and the obstacles (through an occupancy grid).

Taken from the referenced paper

Their analysis and performance is indeed good for geometrical or 2D problems but they have just written some qualitative terms for higher dimensions. I was not sure about it's working on planning problems for something like HERB's arm. So, here is an excellent tutorial that I followed for getting a deeper insight on VAE (Link). I will just give the gist of this over here, maybe for a quick recap.

Essentially, it twist, stretch or turn some normal distributions to generate some unknown distributions. This seems magical, right? But the key idea is that any distribution in d dimensions can be generated by taking a set of d variables that are normally distributed and mapping them through a sufficiently complicated function. I am yet to understand the proof for this.



The standard VAE model represented as a graphical model

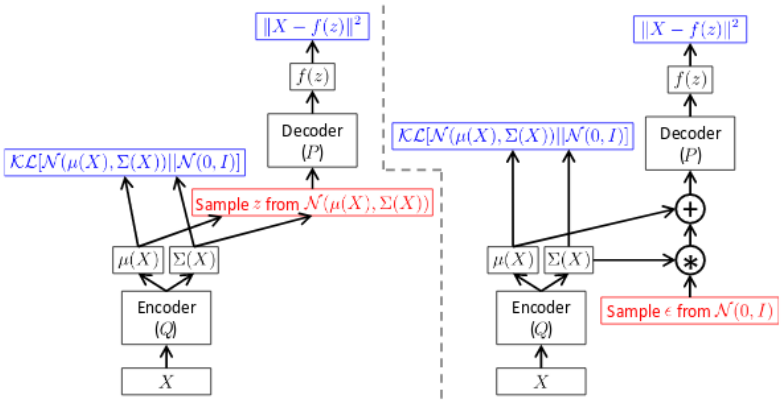
Now, given the distribution z, we want to maximize the probability or likelihood of generating each X that belongs to our dataset. This problem in itself is very difficult, even if we assume some hypothetical deterministic function to relate 'z' to 'X'.

$$P(X) = \int P(X|z;\theta)P(z)dz.$$

One approach could be to sample large no of z and calculate these values for every X and get a computable formula. But with increase in dimension of z, computation time gets affected badly. Moreover, if we notice then practically most of the z values that we are gonna sample this way will have nothing to do with our X, they may be pointing to some random figures, thus wasting the whole computations done. Here, comes the encoder part of VAE. we want to sample only those z that are likely to produce X. Thus we need some new function Q(z|X) that would give a distribution of promising z values. The usual choice is to take Q(z|X) as a N(z|μ, Σ), where μ and Σ are deterministic (can be learned from the model). Now, the core equation of this VAE looks something like this:

$$\log P(X) - \mathcal{D}\left[Q(z|X)\|P(z|X)\right] = E_{z\sim Q}\left[\log P(X|z)\right] - \mathcal{D}\left[Q(z|X)\|P(z)\right]$$

Here, the second term on LHS and last term on RHS are just KL divergence between two distributions. Now, if we consider the second term on RHS then essentially we are just trying to match the two gaussian distribution.

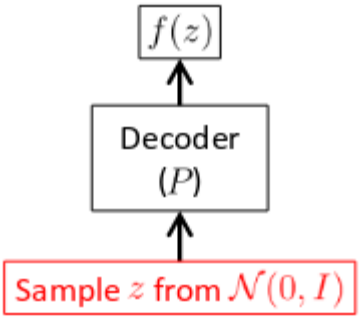


VAE Architecture. Model on right side is with "reparametrization trick"

Essentially, if we see the model on left, we can't back-propagate because of sampling 'z' in the middle which is a non-continuous operation. Thus instead of sampling z directly from Encoder, we sample some ε from a normal distribution and write z = μ + sqrt(Σ)*ε . Thus coming with the final formula:

$$E_{X\sim D}\left[E_{\epsilon\sim\mathcal{N}(0,I)}[\log P(X|z=\mu(X)+\Sigma^{1/2}(X)*\epsilon)]-\mathcal{D}\left[Q(z|X)\|P(z)\right]\right].$$

At test time we just sample 'z' from standard normal distribution and use the decoder to generate 'X'.



Taken from the referenced article

So, this was kind of an introduction to VAE, I would be continuing with the results and some of the generated plots in the articles to follow.