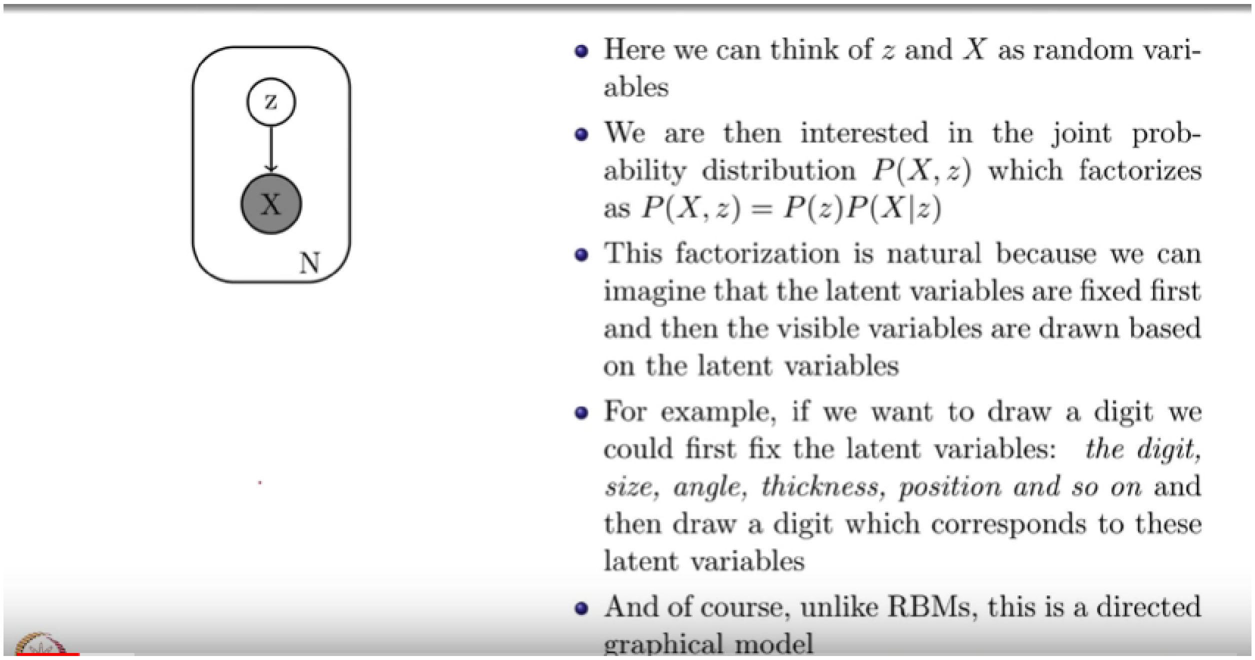
Lecture 20.3
Variational Auto encoders:
The Graphical model perspective

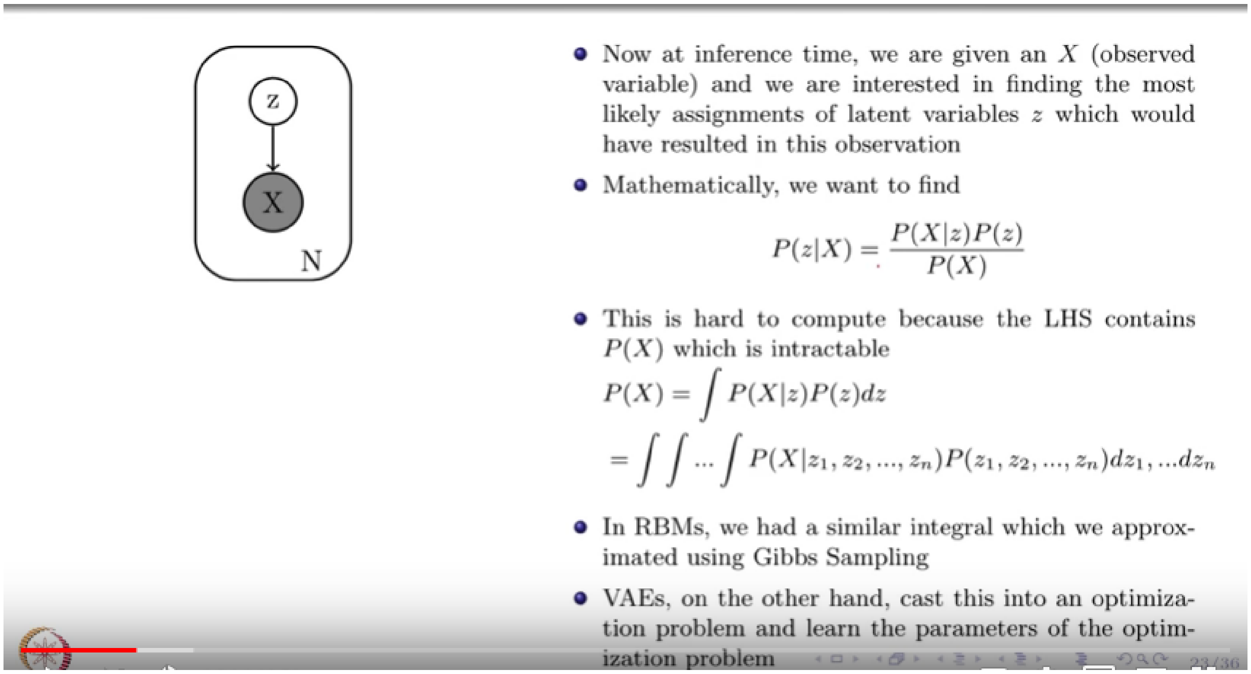
Refer Slide Time :(0:14)



- Here we can think of z and X as random variables
- We are then interested in the joint probability distribution $P(X, z)$ which factorizes as $P(X, z) = P(z)P(X|z)$
- This factorization is natural because we can imagine that the latent variables are fixed first and then the visible variables are drawn based on the latent variables
- For example, if we want to draw a digit we could first fix the latent variables: *the digit, size, angle, thickness, position and so on* and then draw a digit which corresponds to these latent variables
- And of course, unlike RBMs, this is a directed graphical model

So here, we can think of Z and X as random variables, of course. And the graphical model, it comes from this factorization, what I want is actually a joint distribution of X and Z. And the natural factorization that I can chooses, the Joint Distribution factorizes as P of Z into P of x given Z. Why do I call this natural? What is the ordering that I have assumed? That I first, sample from the, latent space. Once I have sample from the latent space, then I sample from the, visible space. Right? And that makes sense? It does make sense. Right? For example, if you want to draw these M missed digits. Okay? Now the M missed digits that, you have been dealing with, they have, several variations. Right? So, it matters first of all, what is the digit that you want to draw? Whether it's 0, 1, 2, 3 and so on. So that's a latent variable. Right? If you're not given the variables and if you're not given the labels and for the entire discussion on graphical models, we have assumed that, we have not given the labels. Right? So what are the little variables? It could be the digit; it could be the size of the digit, whether you want to draw a small line or a big nine, the angle at which you want to draw it. Right? Some people write a bit slanted and so on. The intensity with which you want to write it, either you want to write it as a very bold nine or a very thin looking line and so on. So once you fix these latent variables, then it makes to start talking about the instantiation of that. Or what would the visible variables look like? Without fixing these latent variables, it does not make sense to start talking about okay, I want to produce M missed digits. Right? So that's why, in even in the case the other example that, we are taking about sunny beaches and mountains and so on. It makes sense that you first decide, what kind of image you want to sample? And then you sample from, based on that, latent decision which have made. Okay? Fine. And what kind of a graphical model is this? Has compared to RBMs? RBMs were, I am just hearing some noise, undirected graphical models. And what kind of a graphical model is this? Directed. Right? Because, you have assumed this dependency from Z to X. Okay? You have assumed a direction in the dependency relationship. Okay?

Refer Slide Time :(2:24)



- Now at inference time, we are given an X (observed variable) and we are interested in finding the most likely assignments of latent variables z which would have resulted in this observation
- Mathematically, we want to find

$$P(z|X) = \frac{P(X|z)P(z)}{P(X)}$$

- This is hard to compute because the LHS contains $P(X)$ which is intractable

$$P(X) = \int P(X|z)P(z)dz$$

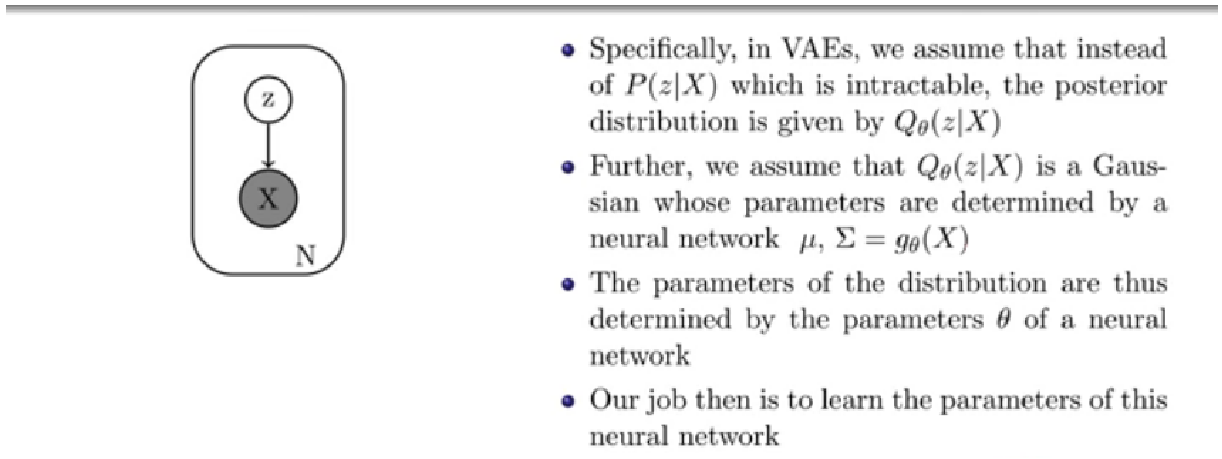
$$= \int\int ... \int P(X|z_1, z_2, ..., z_n)P(z_1, z_2, ..., z_n)dz_1, ...dz_n$$

- In RBMs, we had a similar integral which we approximated using Gibbs Sampling
- VAEs, on the other hand, cast this into an optimization problem and learn the parameters of the optimization problem

23/36

Now, given this setup or this graphical model perspective. Now, we can talk of things that, we are interested at, inference time, it’s I have learned this joint distribution. And as usual, the learning part will always come at the end; we’ll first assume that we have learned this joint distribution. And now let’s see, what is it that we are interested in at inference time? So what is it that very interested in inference time? One of the things that very interested in is, I give you an X and you give me the latent representation. Right? So in terms of, probabilities, can you tell me what is it that I'm interested in? Probability of z given x. Right? This is what I'm interested in and I can write it as this. Okay? Whatever you see on the right hand side? Is there a problem with this? Of course assumes you have learned the parameters. Is there a problem with this? Which term is a problem here? P of the denominator. The denominator again this, massive integral weight, so the denominator is nothing but this, integral of the new numerator. Right? And here's, where add this, so this integral in turn is an integral over many, many variables, its Z1 to Z n. So this is again intractable and we come back to the same situation that, we are interested in computing some quantity, but that requires this expectation and that expection, expectation is intractable. Because, you have a large number of variables. Okay? In RBMs, how did we get rid of this expectation? By doing Gibbs Sampling. Right? So, that's one way of tracking it or rather sidestepping it. In Variational Autoencoders, we're going to do something different. We are going to cast this into an optimization problem and then, try to learn the parameters of that optimization. Right? So that's a different way of approaching this problem, both RBM’S and variation Autoencoders deal with the same problem that, you have this in, intractable expectation or integral, but there are different ways of dealing with it. Okay?

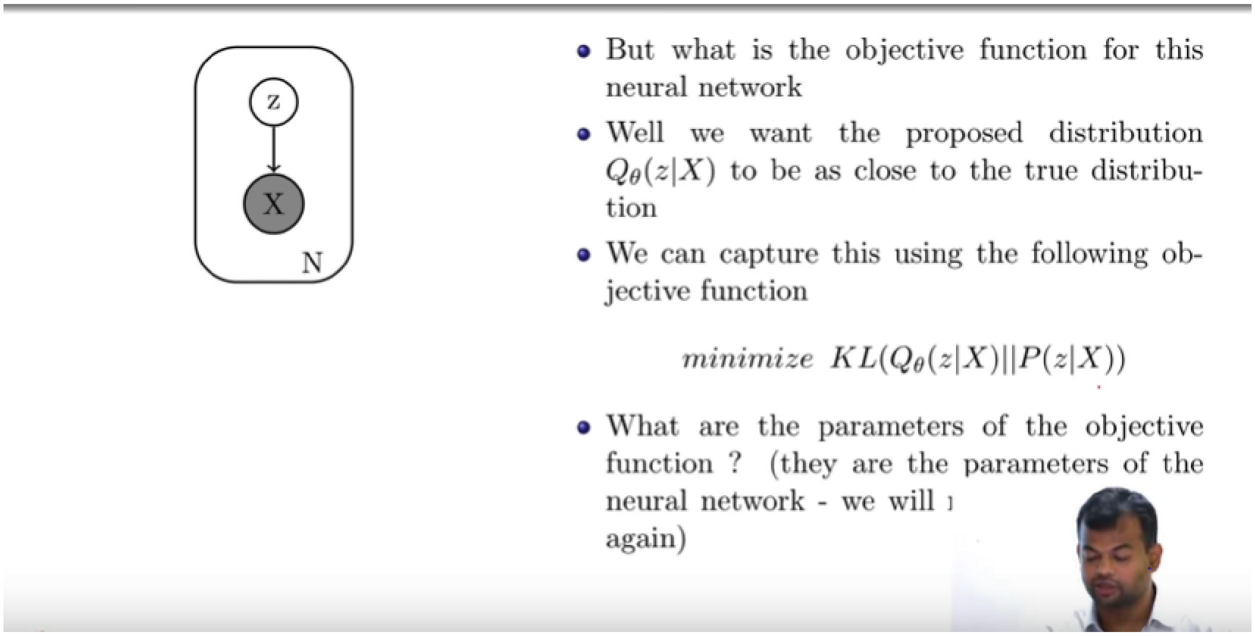
Refer Slide Time :(4:29)



- Specifically, in VAEs, we assume that instead of $P(z|X)$ which is intractable, the posterior distribution is given by $Q_\theta(z|X)$
- Further, we assume that $Q_\theta(z|X)$ is a Gaussian whose parameters are determined by a neural network $\mu, \Sigma = g_\theta(X)$
- The parameters of the distribution are thus determined by the parameters θ of a neural network
- Our job then is to learn the parameters of this neural network

So in these, what we do is, we assume that, instead of PZ given X, which is the actual thing that we should have computed. And it is intractable, the posterior distribution that we are interested in is some q theta Z given X. this is what I’m going to assume? Right? So I am saying that, I don't know what P's had given X's, I can't even compute it, but I am saying that, there is another Q Z given X, which has parameters theta. And now, I’m going to set up an optimization problem. What should be the goal of that optimization problem? What are the parameters of that object optimization problem? Theta. Okay? That’s one thing is clear, what should be the objective function? We are dealing with two distributions; I know, there's a two distribution, which I can't compute, I am proposing an arbitrary distribution with some parameters, what is it that I would expect from this distribution? It should be, as close and how do you say this, in mathematical terms, I would want to minimize the KL divergence, between these two distributions. Okay? Now, the first thing I'm going to do is, again I'm going to assume a family for Q, I am going to assume that Q comes from a Gaussian family or the normal distribution. And the parameters are mu and Sigma. And these parameters in themselves are, expressed as some other parameters. As a function of some other parameters and a function of the input of course. Because, this is the given quantity same, same thing as what we saw in the neural network perspective, nothing different here. And now, the parameters of the distribution, our parameters are explained in terms of parameters of some other function and our job is now to learn, these secondary parameters. Right? Okay? And again, will eventually reach the same destination that, these parameters are going to be the parameters of some neural network. Okay? But, in an abstract way, this is what I am going to write it as. Right? So what have done is, I couldn't compute P, I have proposed a Q, which has certain parameters, these parameters I am going to express, as function of some other parameters, I am going to set up an objective function or an optimization problem. And the optimization problem is going to be with this set of secondary parameters. Okay? Fine.

Refer Slide Time :(6:49)



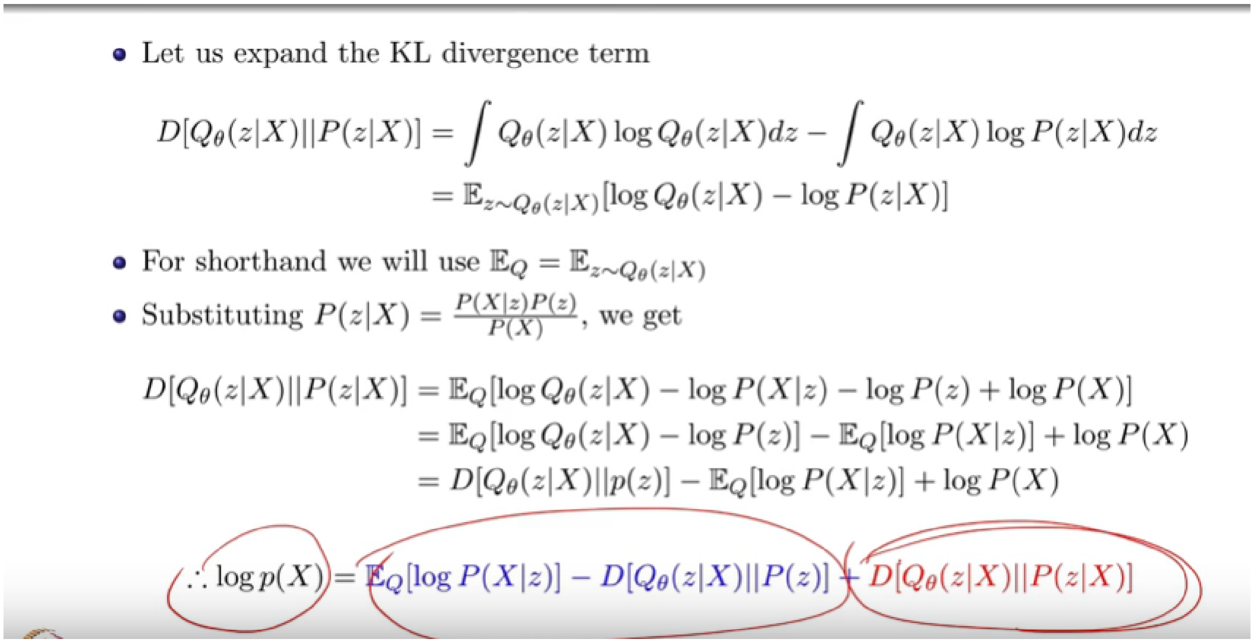
- But what is the objective function for this neural network
- Well we want the proposed distribution $Q_\theta(z|X)$ to be as close to the true distribution
- We can capture this using the following objective function

$$minimize\ \ KL(Q_\theta(z|X)||P(z|X))$$

- What are the parameters of the objective function ? (they are the parameters of the neural network - we will ı again)

And now, what's the objective function? KL evidence. Right? So KL divergence between, the proposed distributions which is Q theta. And the two distribution which is P. Okay? This is what I want to minimize? And what are the parameters of the objective function? Every one, theta. Right? So, they're the parameters of the neural network and we'll come back to this. Okay?

Refer Slide Time :(7:12)



- Let us expand the KL divergence term

$$\begin{aligned} D[Q_\theta(z|X)||P(z|X)] &= \int Q_\theta(z|X)\log Q_\theta(z|X)dz - \int Q_\theta(z|X)\log P(z|X)dz \\ &= \mathbb{E}_{z\sim Q_\theta(z|X)}[\log Q_\theta(z|X) - \log P(z|X)] \end{aligned}$$

- For shorthand we will use $\mathbb{E}_Q = \mathbb{E}_{z\sim Q_\theta(z|X)}$
- Substituting $P(z|X) = \frac{P(X|z)P(z)}{P(X)}$, we get

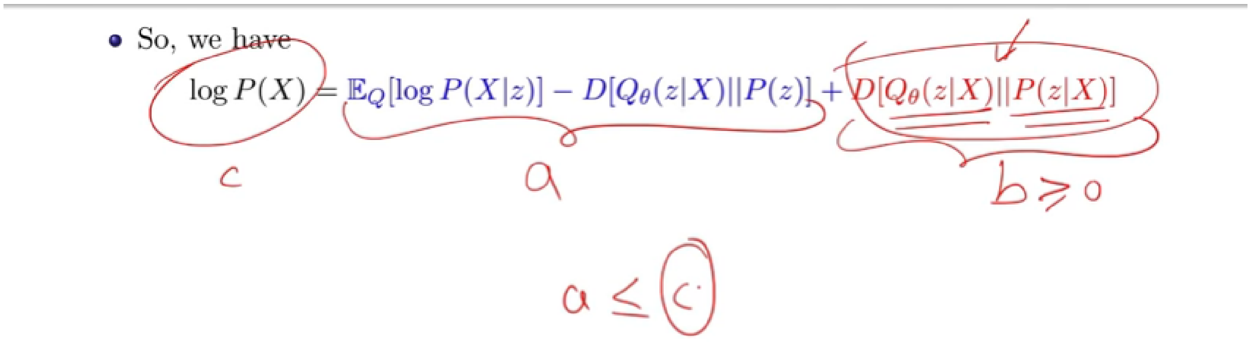
$$\begin{aligned} D[Q_\theta(z|X)||P(z|X)] &= \mathbb{E}_Q[\log Q_\theta(z|X) - \log P(X|z) - \log P(z) + \log P(X)] \\ &= \mathbb{E}_Q[\log Q_\theta(z|X) - \log P(z)] - \mathbb{E}_Q[\log P(X|z)] + \log P(X) \\ &= D[Q_\theta(z|X)||p(z)] - \mathbb{E}_Q[\log P(X|z)] + \log P(X) \end{aligned}$$

$$\therefore \log p(X) = \mathbb{E}_Q[\log P(X|z)] - D[Q_\theta(z|X)||P(z)] + D[Q_\theta(z|X)||P(z|X)]$$

So now, first let's what we will do is? I've just read it in the scale divergence and the scale; I am just going to call it as,' D'. It just stands for divergence. Right? So, we are going to work with the scale divergence and try to expand on it, it does see what it looks like. Okay? So what's the formula for KL divergence?

It’s not, is it symmetric, asymmetric, asymmetric. Okay? So that's good. So, the order in which you tell me matters. Right? So it's, dash log dash minus dash log dash, so and of course integral. Now, I'm also confused, no it's, not that. So it's Q log Q - Q log P. Okay? And the integral again this, integral is intractable; whenever we have an integral what do we replace it by? What do we replace it by? Summation, expectation. Right? We just write it as, an expectation. So we can write this as, so inside the expectation, I have these two quantities, just so that you understand and outside I have just written it as, an expectation with respect to this distribution, which is exactly the first term and both these integrals is, this conversion from the integral to the expectation clear. Right? Everything is fine with that, I why do I see blank faces. Okay? And now since, this is a bit tedious to write, I'm just going to use a shorthand Q for this. Okay? So this entire quantity, whenever you see capital Q in the expectation it means, Z coming from the distribution Q theta of Z given X, we're just saying it itself is tedious. Okay? So and so that's what I'm going to do for this guy. And now the second thing, I’m going to do is, I'm going to make write P of Z given X as, the standard thing, I'll just write it as the Bayes rule. Okay? So now, what will happen inside? What, what will be the terms that I will get inside? Of course this term will remain as it is, because I am not touching it. what about this term? Log of minus log P of x given Z, everyone please, I’m assuming it's too easy that's why you’re just, offended that I'm asking you to do this. Okay? So that's what will happen. Right? So, this term will get replaced by these two terms, so the two numerators and the one denominator. Right? And the signs are getting adjusted accordingly. Okay? So Okay? I had two terms to deal with now; I have four terms to deal with. Now, one of these terms it looks like, I can take it out of the expectation, which one? Which is the term? Which I can take out of the expectation? And there is not a simple question. So, I will wait for everyone to answer, the expectation is with respect to which random variable Z. So which of these terms can I take out of the expectation, log of, P of X.? Right? So that’s the one, which I'm going to take out. Okay? And I have this quantity; I have this quantity and this quantity. Okay? What’s the first quantity? Yeah, he knows why I did that. It’s the, this what I meant is this, looks very similar to this. And which was actually derived from here, so now can you tell me what it is? KL divergence between, QZ given X and P of Z. Okay? The second term I'll leave it as it is and the third term also I'll leave it as it is. Okay? And now, I just rearranged something’s, I'll keep log P of X on one side why? Because, because, what's the recipe? Maximize D, so that's the term which I care about, I'll keep it on one side. Okay? And then, I have this blue term and then I have this red term. Okay? If I take the red term on the other side, what will I get? The place where we left off, when we are doing the neural network perspective. Right? So we had said that, we want to maximize the log likelihood of the data and minimize the KL divergence oops okay, doesn’t matter, we can anyways continue. So, we have this log of P X on one side and you have these two terms, on the other side. Okay?

Refer Slide Time :(11:42)



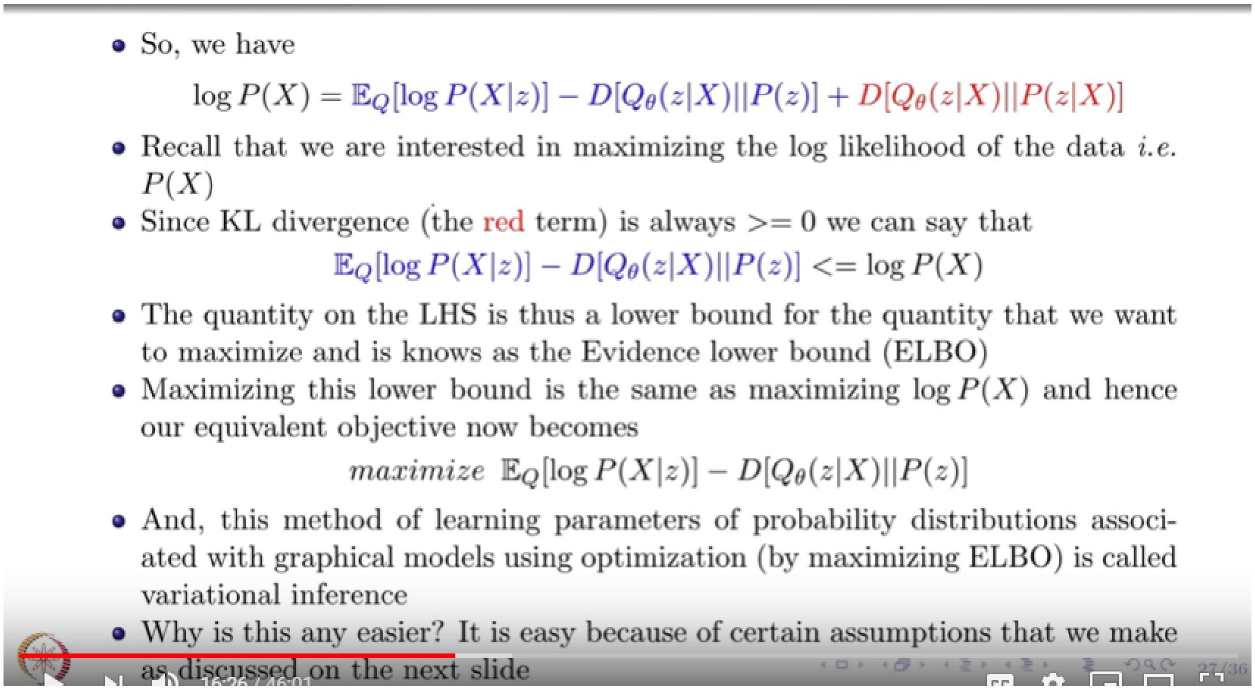
- So, we have

$$\underbrace{\log P(X)}_{c} = \underbrace{\mathbb{E}_Q[\log P(X|z)] - D[Q_\theta(z|X)||P(z)]}_{a} + \underbrace{D[Q_\theta(z|X)||P(z|X)]}_{b \geqslant 0}$$

$$a \leq c$$

Now, what is the red term actually? This was the starting point. Right? This is the quantity, which I said we want to minimize. I said that, there is a true distribution, there is a proposed distribution parameterized by theta and I want to, minimize. Minimize the difference between these two. Okay? That’s what my starting point was? The other way to look at it is that. And now, what's the property of KL divergence? It’s always greater than 0, less than 0, greater than equal to 0, less than equal to 0. What is it greater than equal to 0? Okay? So, now this is the quantity that I care about. Okay? let me just call this as,’ A’. and let me just call this as,’ B’. So, the quantity that I care about, let me just call it a C, there is actually a plus B and B is. Okay. So what can I tell about the relation between a and C? there may be things you can say about it, I want in terms of greater than, less than, equal to, C is greater than on equal to a. Right? Or other A is less than equal to C, just to give a different answer. So what does that mean, actually in terms of lower bounds, upper bounds and so on? Remember this is the quantity that, I want to maximize. Now, I have another quantity, which I know is always less than or equal to this quantity. Or other I know that this quantity is always, greater than equal to that quantity. Now think of it this way, I want to maximize C, I know I cannot maximize C why? Because it's, computed computing it is interactively, do you have this integral over integral and so on. I cannot compute that, but I have this another quantity, which I know, that C is always going to be greater than that quantity, should I take this other quantity and I keep maximizing it, what would happen to see? You will also get maximized. Do you get that, everyone gets this, so instead of maximizing C, what can I maximize? A, does that make sense. Okay? Why do this god knows? But, at least by the promisor I did you can do this. Okay?

Refer Slide Time :(14:11)



- So, we have

$$\log P(X) = \mathbb{E}_Q[\log P(X|z)] - D[Q_\theta(z|X)||P(z)] + D[Q_\theta(z|X)||P(z|X)]$$

- Recall that we are interested in maximizing the log likelihood of the data *i.e.* $P(X)$
- Since KL divergence (the red term) is always $>= 0$ we can say that

$$\mathbb{E}_Q[\log P(X|z)] - D[Q_\theta(z|X)||P(z)] <= \log P(X)$$

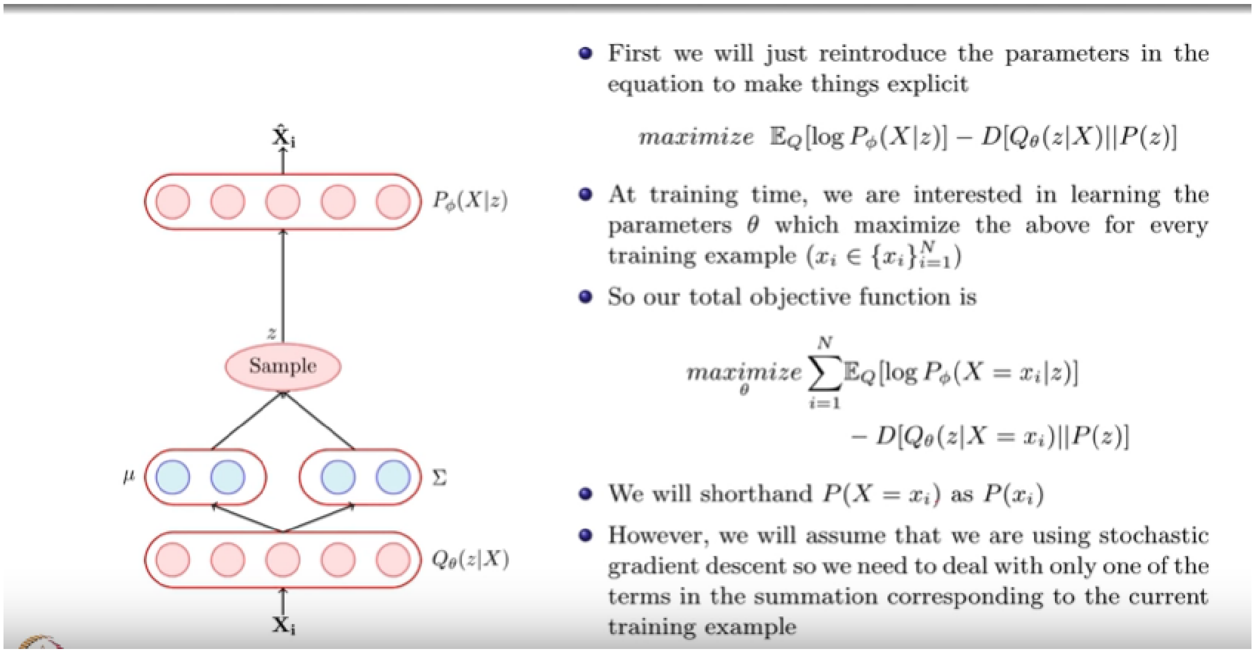
- The quantity on the LHS is thus a lower bound for the quantity that we want to maximize and is knows as the Evidence lower bound (ELBO)
- Maximizing this lower bound is the same as maximizing $\log P(X)$ and hence our equivalent objective now becomes

$$maximize\ \ \mathbb{E}_Q[\log P(X|z)] - D[Q_\theta(z|X)||P(z)]$$

- And, this method of learning parameters of probability distributions associated with graphical models using optimization (by maximizing ELBO) is called variational inference
- Why is this any easier? It is easy because of certain assumptions that we make as discussed on the next slide

So, we are interested in maximizing the log likelihood of the data. And we have this relation that the log likelihood of the data or this is wrong, I why do I always make mistakes on the most crucial point, oh no, no this is correct. Okay?, Okay? The red term is gone; I thought the red term became blue or something. Okay? This is fine, so this is again the same thing CAN. So I can't deal with C, so the quantity on the left hand side is actually a lower bond, for the quantity on the Right hand side. so if I can't deal with this guy, let me just deal with the lower bound, if I can maximize the lower bound is, the same as maximizing this guy, is that clear? Okay? So that's what I'm going to do, I am now going to come up with an equivalent objective function, which is to maximize this blue term, that we had. Okay? And the blue term, has one KL divergence and another expectation term. So, I still need to tell you whether this is any easier to deal with, what are the parameters of this distribution? Or this sorry, what are the parameters of this objective function? I still need to tell you all these things. Right? But, for now we have arrived at a certain objective function, starting from, I would say legitimate and reasonable and well thought of well principle step, so we have not done any random things anyway, you have just started with, what you would like to do? And gave a justification for doing that is, equivalent to doing this. Okay? Fine. So this, quantity Right? There is nothing unique to variation Autoencoders, this is used in a lot of graphical models, this is known as the evidence lower bound, because it gives you, a lower bound, on the probability of the evidence that you are seeing. Okay? And this method of optimizing this blue term, instead of the term that you actually care about, which is log P of X is known as, barrister inference. And hence, these auto encoders that we are looking at are known as, please show some enthusiasm, variation Autoencoders. Okay? Of course the question is why is this any easier? Than working with the original objective function? Right? and this would be easier, because of certain assumptions that we are going to make, when you're working with graphical models, let's deal with the assumptions and approximations, don't cringe every time I use the word assumption an approximation. Okay? Fine.

Refer Slide Time :(16:31)



- First we will just reintroduce the parameters in the equation to make things explicit

$$maximize \;\; \mathbb{E}_Q[\log P_\phi(X|z)] - D[Q_\theta(z|X)||P(z)]$$

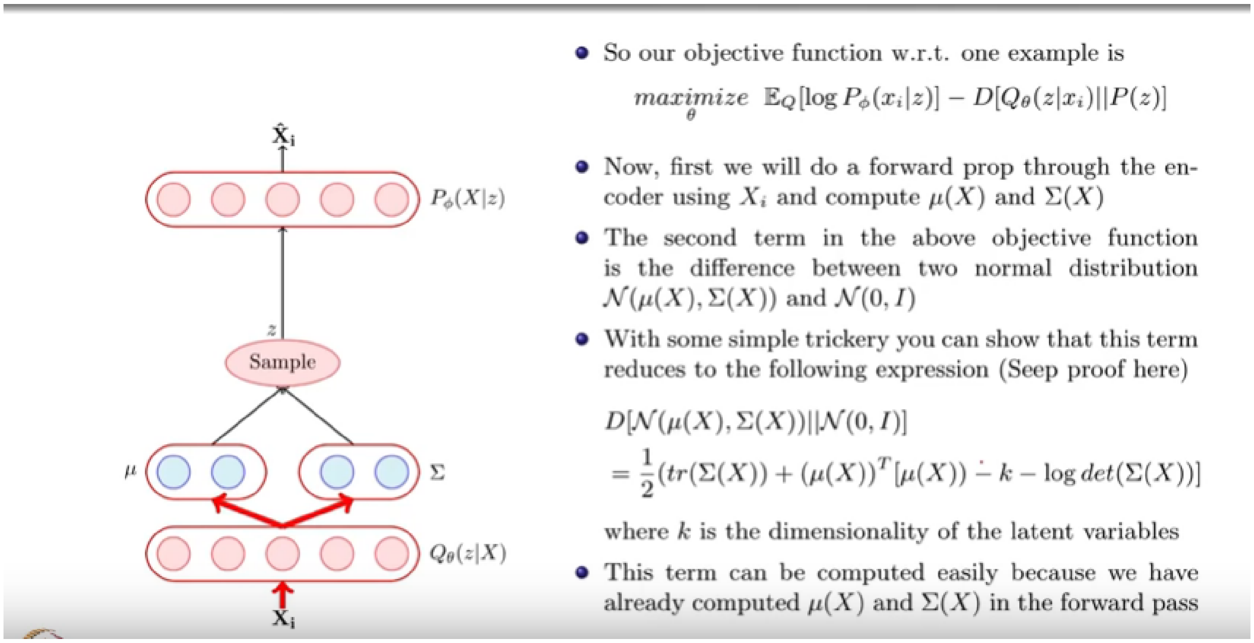
- At training time, we are interested in learning the parameters θ which maximize the above for every training example ($x_i \in \{x_i\}_{i=1}^N$)
- So our total objective function is

$$\underset{\theta}{maximize} \sum_{i=1}^{N} \mathbb{E}_Q[\log P_\phi(X = x_i|z)] - D[Q_\theta(z|X = x_i)||P(z)]$$

- We will shorthand $P(X = x_i)$ as $P(x_i)$
- However, we will assume that we are using stochastic gradient descent so we need to deal with only one of the terms in the summation corresponding to the current training example

So, first let us just reintroduce all the parameters. Right? So this was our objective function and now, we have reintroduced the parameters. So, Q was parameterize by theta and P of x given Z was parameterize by Phi, nothing new here, this is the same that, we had introduced even in the neural network perspective. So I am back, I am trying to kind of now merge the two, so from the neural network perspective, I had reached a certain objective function, you can go back and check that, we have reached an equivalent objective functions, starting from the graphical model perspective. And now, the only job left is one is that, this optimization function is with Theta Phi, we need to add this. Okay? And the other thing is to show that, this is Tractable, fine. So, now let's get into training. So, at training time, we are interested in learning the parameters of theta and Phi, which maximize the above for every training example, given to us. Right? Foreach of the exercise, we are interested in maximizing this quantity. Okay? So that total objective function is just going to be sum over, all these individual objective functions, is that fine. Okay? But, now my favorite algorithm is stochastic gradient descent, so what do I do in stochastic gradient descent? What's my loss function there? One of the terms in the summation. Right? So Cassity gradient descent, I just have one training example, I compute the loss for that and just back propagate and that's just the same as, saying that, I am going to deal with only, one of these terms in the summation, at any given point of time is that. Okay? Fine. Okay? So and of course we will shorthand this, this just a side note, so we're going to assume that, we are using stochastic gradient descent that means, at any point, I just need to deal with one of the terms in the summation.

Refer Slide Time :(18:18)



- So our objective function w.r.t. one example is

$$\underset{\theta}{maximize}\ \mathbb{E}_Q[\log P_\phi(x_i|z)] - D[Q_\theta(z|x_i)||P(z)]$$

- Now, first we will do a forward prop through the encoder using X_i and compute $\mu(X)$ and $\Sigma(X)$
- The second term in the above objective function is the difference between two normal distribution $\mathcal{N}(\mu(X), \Sigma(X))$ and $\mathcal{N}(0, I)$
- With some simple trickery you can show that this term reduces to the following expression (Seep proof here)

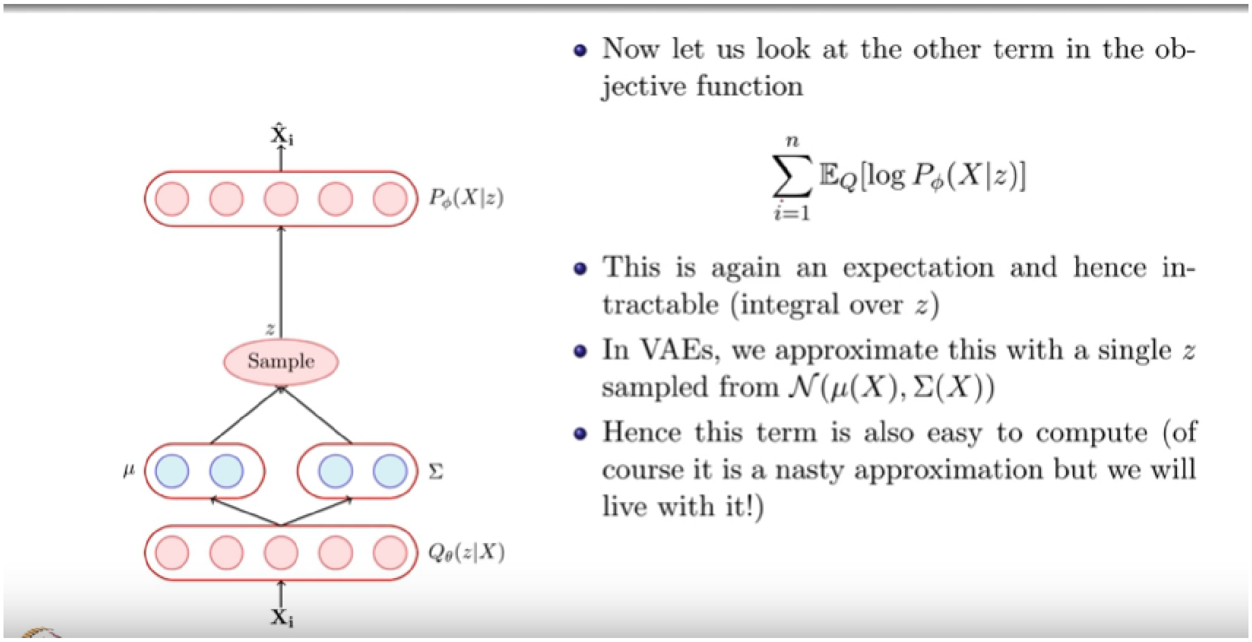
$$D[\mathcal{N}(\mu(X), \Sigma(X))||\mathcal{N}(0, I)]$$
$$= \frac{1}{2}(tr(\Sigma(X)) + (\mu(X))^T[\mu(X)) \dot{-} k - \log det(\Sigma(X))]$$

where k is the dimensionality of the latent variables

- This term can be computed easily because we have already computed $\mu(X)$ and $\Sigma(X)$ in the forward pass

So, my effective objective function is going to be, just one of the terms, I basically got red of the summation and I've just have this, one term that I need to focus. Okay? Now, first now you have given me this example X i, I should first be able to compute the loss, if I'm able to compute the loss, back propagation is not my problem like that I can Okay? So, first let me try to compute the loss. so let's, look at this term first, so Weill be looking at that term, but, before that the moment you give me a sample X I, what am I going to do? I am going to pass it through the encoder and compute, what mu X and Sigma X? Is there something wrong here? And let's assume Sigma also is there something wrong here, was I interested in the means of Z or X, what does the encoder predict? P of Z given X, the parameters of P of Z given X. Why I have written as mu of X? It’s a function of X, so please don't get confused, there it is not the mean of X, I am just saying that the correct way of noting this, would have been mu Z, which is a function of X. Right? So mu Z of X. but, that's too cumbersome and it will become hard to read as you go ahead, so remember that when I say mu of X, I just mean that compute the mean of Z given X, which is a function of X is everyone clear with this notation, if you’re not you will get totally lost, for the rest of your life. Okay? Is that clear. Okay? so that's what I mean, mu is a function of X and nu is actually a parameter of the distribution Z given X. so the moment I give you X, with whatever is your current configuration of the parameters is, a simple feed-forward neural network, you can give me, mu and Sigma. Okay? Now, I can do this. So now, the second term here was actually the KL divergence, between these two distributions. What was this distribution the second one P of Z? What did we assume? Standard normal distribution. Okay? So the second term is actually, the KL divergence between two normal distributions, one being this. And the other being this, do you guys, know a formula for the KL divergence between two stand, two normal distributions, I want everyone to say yes, why do you know that? Or you all did it in the assignment. Okay? So, I hope this is the formula that you arrived at, if not then you can be sad after the lecture is over, but Right now, please focus is that fine. Okay? So, the difference between two normal distributions can be computed, as a closed form and in particular, if one of the distributions happens to be zero I, this is the answer that you get. Is this quantity easy to compute? What is this quantity? Is it a scalar, a vector, matrix, a tensor, scalar is it easy to compute. Okay? Let’s look at, each of these terms. Okay? You’re looking at the trace of Sigma X, straight forward, you are looking at the dot product between two vectors, I'll not tell you what K is, although it's written on the slide. And then they are looking at the determinant. So, all of this is straightforward to compute. Okay? So at least, one term of the objective function is easy to compute, at least that much, we have achieved by transforming all that, circus and coming to this new objective function. Right? So certainly I've shown you that, one term is very easy to compute. And K is nothing but the dimensional reality of the latent variables, the number of latent variables that you have. Right? And that's, I hope this is the closed form solution that, you came up with in your assignment. Okay? Is that fine. So given an X, I can compute mu and Sigma, plug mu and Sigma into this formula and I am done. I have computed the second term of the last function, everyone is fine with this. Okay? Now, this is the second part is what you are not going to be fine. Right? But, what can I do? So this term can be computed easily, once you have done the forward pass.

Refer Slide Time :(22:28)



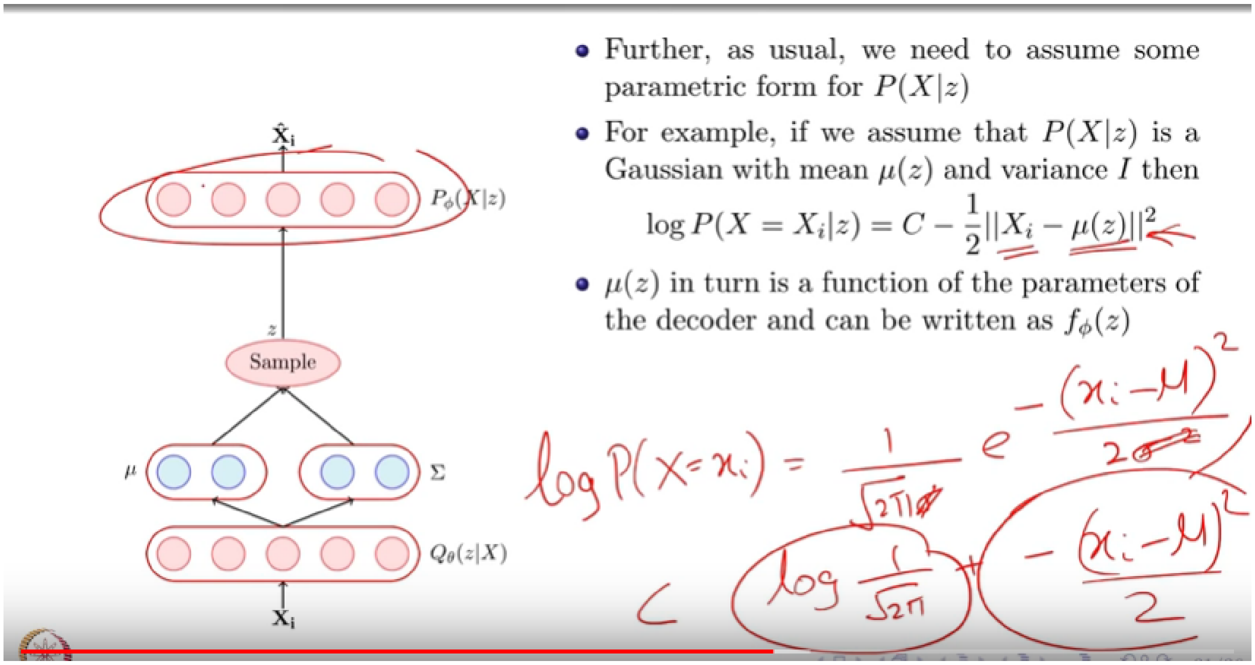
- Now let us look at the other term in the objective function

$$\sum_{i=1}^{n} \mathbb{E}_Q[\log P_\phi(X|z)]$$

- This is again an expectation and hence intractable (integral over z)
- In VAEs, we approximate this with a single z sampled from $\mathcal{N}(\mu(X), \Sigma(X))$
- Hence this term is also easy to compute (of course it is a nasty approximation but we will live with it!)

Now, let's look at the other term in the objective function, any guess, what I'm going to do. This term is supposed to be computed. Okay? So, this was not this, sorry. You already have one nasty integral there; we don't need the other summation. Right? This was summation over all the data points, but we don't need that. Right? Because, I just dealing with one data point. So this is the expectation, with respect to Q, the distribution Q. how am I going to compute this, this is as bad as anything it, this is again some nasty integral is going to come into play. I mean, forgotten what's there on the next plane is? this what you all expected, this expectation has to be computed over all possible values of Z, I'm going to replace the expectation by a point estimate, something similar we did in RBM'S. Hence, I had that slide, which said these are the characteristics of RBM's. So it's almost, you can see that we are coming back to similar assumptions, but using them in very different context. Right? So, I'm going to replace, this entire expectation, using a sim, single Z, drawn from the distribution and mu comma Sigma and I have already computed mu comma Sigma in the forward pass, is that fine. So I've computed mu comma Sigma, I am going to sample as Z from there and I'm just going to estimate this expectation, using this single point estimate is that fine. Hence, computing the first time is also very easy. and of course, I didn't you did a nasty approximation, but we'll just live with it just because, that's going to be a part of the game, once you’re dealing with graphical models, which have a large number of variables, you'd have to do some kind of an approximation or an assumption. So here, we are making this approximation, which is a point estimate. Okay?

Refer Slide Time :(24:12)



- Further, as usual, we need to assume some parametric form for $P(X|z)$
- For example, if we assume that $P(X|z)$ is a Gaussian with mean $\mu(z)$ and variance I then

$$\log P(X = X_i|z) = C - \frac{1}{2}||\underline{X_i} - \underline{\mu(z)}||^2$$

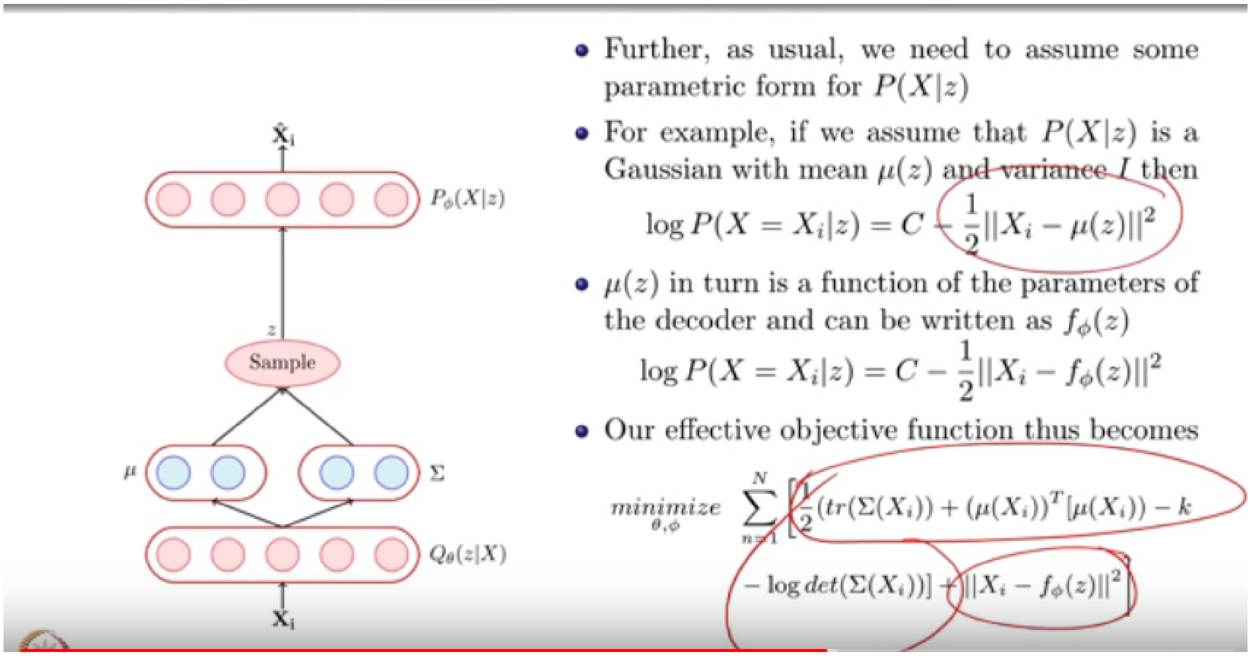
- $\mu(z)$ in turn is a function of the parameters of the decoder and can be written as $f_\phi(z)$

$$\log P(X = x_i) = \frac{1}{\sqrt{2\pi\cancel{\sigma}}} e^{-\frac{(x_i-\mu)^2}{2\cancel{\sigma}^2}}$$

$$C \quad \log\frac{1}{\sqrt{2\pi}} + -\frac{(x_i-\mu)^2}{2}$$

So, that is one thing. The second is as usual we are going to assume some parametric or some family for P of x given Z. and again in variation auto encoders, we are going to assume that P of x given Z is actually, a normal distribution. So, if that's the case, what does this term actually boil down to? So why am I computing this? Because, I'm going to replace this by one single value. Right? And that single value is nothing but log of P of X equal to X I, which was my input, given the Z which I have, sampled from the distribution. So this is one quantity, which I need to compute. Now, in the case of a normal distribution, what does it boil down to? You can answer it even though the answer is there, it's log of e raise to minus something, so log in he will cancel out, what’s that – something? X I minus mu. Now, this I'm calling it as mu Z, what does that mean? It’s the funk it's the mean, of the distribution x given Z, which is a function of Z. and that's what the decoder is going to compute. Okay? And this is what? That one point estimate boils down to, how many forget this please raise your hands, not many. So just write it as someone by some constant e raised to minus X I – mu Z, V whole square and I can just do it in one variable case. Right? So, you have the Sigma square and made assume that this is unit variance, so this is 1, it should be 2 Sigma squared. Okay? So what is this? This gives me the probability of X equal to X I, that's what, the normal distribution gives you, how many fail completely lost at this point? All I have done is written the distribution is that such a crime, oh I see, why you lost. Okay? How many of you still don't get this? So, so for the normal distribution. Okay? let's think about, the univariate case P of X equal to X I is given by 1 by square root of 2Pi Sigma, can anyone help me with this, E is to minus X I minus mu of whatever mu, whole square divided by 2 Sigma square, the same formula you can imagine it in a multivariate case also, you should not imagine you should know it, but, if you don't know it just imagine it. Okay? Fine. Now, in fact should I just work with the univariate case? Okay? Not now, whatever, whatever assumed about the variance? Unit variance. Right? So this goes off. Okay? so this is, P of X equal to X I. now, if I take the log of this what will I get? log of 1 by square root of 2 pi plus log of e raised to this, the log in the e will cancel and you will be left with X I minus mu the whole square .okay? By 2. This term I am calling it as a constant and this exactly is the same as this, except that it's in the multivariate case. Now, everyone gets this, so if you have a Gaussian distribution, the log of the Gaussian distribution or the log of the probability is just going to be the square error difference, between your observation and the mean of the distribution, everyone gets this. Okay? Is this term easy to compute, do you know XI? Can you compute the second term? How will you compute this you have sampled Z? It you just pass it through the decoder? The decoder will give you mu of x given Z, as a function of Z and you just take the difference. so you're just taking whatever the decoder predicts that's your Mu, you're taking the difference from the input, I just taking the squared difference, everyone gets that. How many if you don't get this? Some people started raising their hands, how many forget this? Okay? Good.

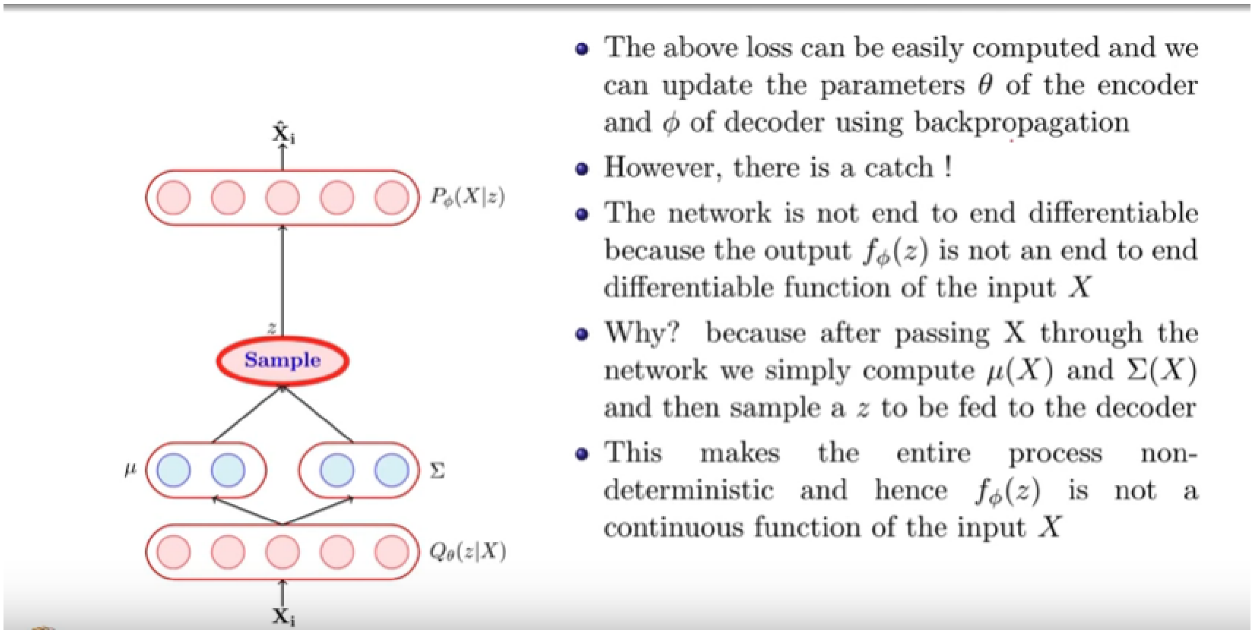
Refer Slide Time :(28:40)



- Further, as usual, we need to assume some parametric form for $P(X|z)$
- For example, if we assume that $P(X|z)$ is a Gaussian with mean $\mu(z)$ and variance I then
$$\log P(X = X_i|z) = C - \frac{1}{2}||X_i - \mu(z)||^2$$
- $\mu(z)$ in turn is a function of the parameters of the decoder and can be written as $f_\phi(z)$
$$\log P(X = X_i|z) = C - \frac{1}{2}||X_i - f_\phi(z)||^2$$
- Our effective objective function thus becomes
$$\underset{\theta,\phi}{minimize} \sum_{n=1}^{N} \Big[\frac{1}{2}(tr(\Sigma(X_i)) + (\mu(X_i))^T[\mu(X_i)) - k$$
$$- \log det(\Sigma(X_i))] + ||X_i - f_\phi(z)||^2\Big]$$

So the first term, so now we have the full loss function. Right? So, this is what we got from the KL divergence term, no not this, this whole thing. And this second term is what we had from that nasty first expectation? Which we approximated by a point estimate? And the point estimate just happened to be this value here. Okay? So, now we have this full loss function, which we know is computable of course it is. And now, what do we do? Go home. Now, what next? You have the loss function, you have the data, you have the by now you should memorize the sequence rate data, model, parameters, objective function, what's missing? What's the algorithm? Sorry, stochastic gradient descent, back propagation, what's the, what's the law, what's the problem with using back propagation here? Can you use back propagation here? Can we do back propagation here? Okay? let's in fact, I will just said we know the data, we know the model, we know the parameters, we know the objective function, write down the model, write down the output as a function of the input, can you do that? Can you write the output as a continuous function of the input, why no? Well we have this sampling step, you cannot write it down as a deterministic function of the input. Right? The sampling brings in this randomness, which brings in this discontinuity, the moment you have a discontinuity, what can you not compute? The gradients. If you cannot compute gradients, gradient descent is off, back propagation is off. Right? So that's still one problem; that we need to solve, we have solved the problem of the loss function, but we still don't know training algorithm. Okay?

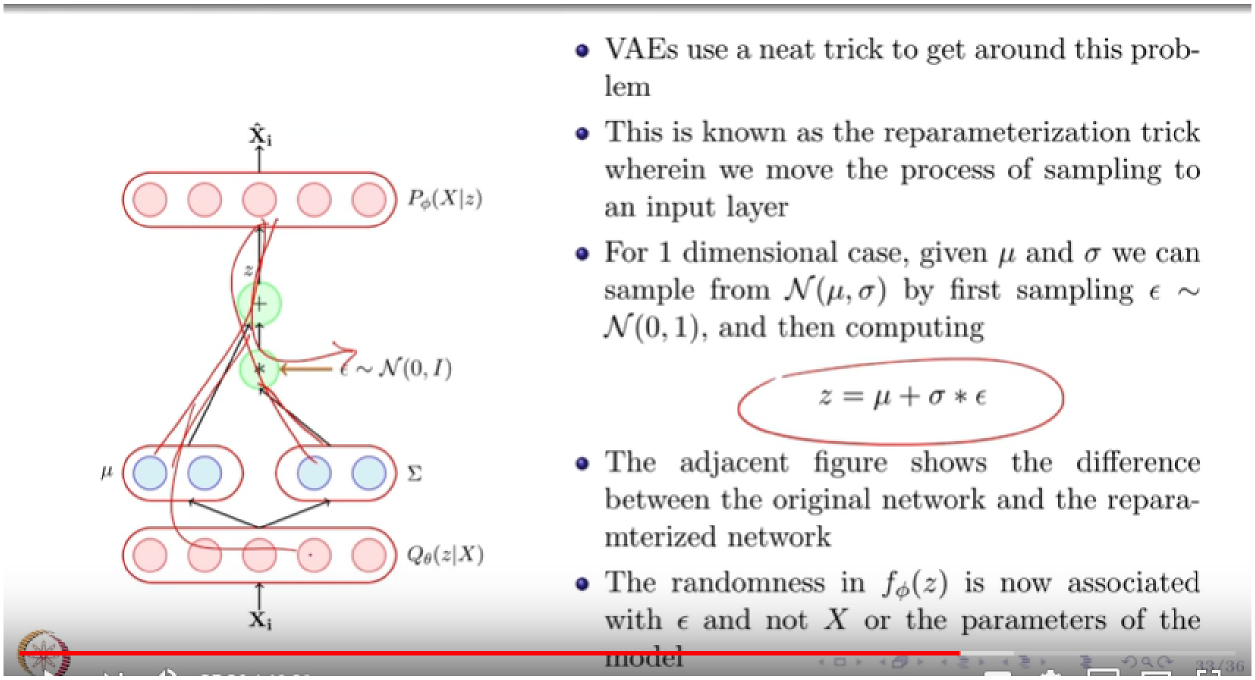
Refer Slide Time :(30:41)



- The above loss can be easily computed and we can update the parameters θ of the encoder and ϕ of decoder using backpropagation
- However, there is a catch !
- The network is not end to end differentiable because the output $f_\phi(z)$ is not an end to end differentiable function of the input X
- Why? because after passing X through the network we simply compute $\mu(X)$ and $\Sigma(X)$ and then sample a z to be fed to the decoder
- This makes the entire process non-deterministic and hence $f_\phi(z)$ is not a continuous function of the input X

So, the last function can be easily computed. And the objective function the optimization problem is with respect to theta and Phi, there's a catch, the cache is that you're using this sampling, you're sampling from the distribution, mu comma Sigma for the variable Z and that brings in this discontinuity. And because, of that you cannot, have the output as a continuous function of the input.

Refer Slide Time :(31:05)



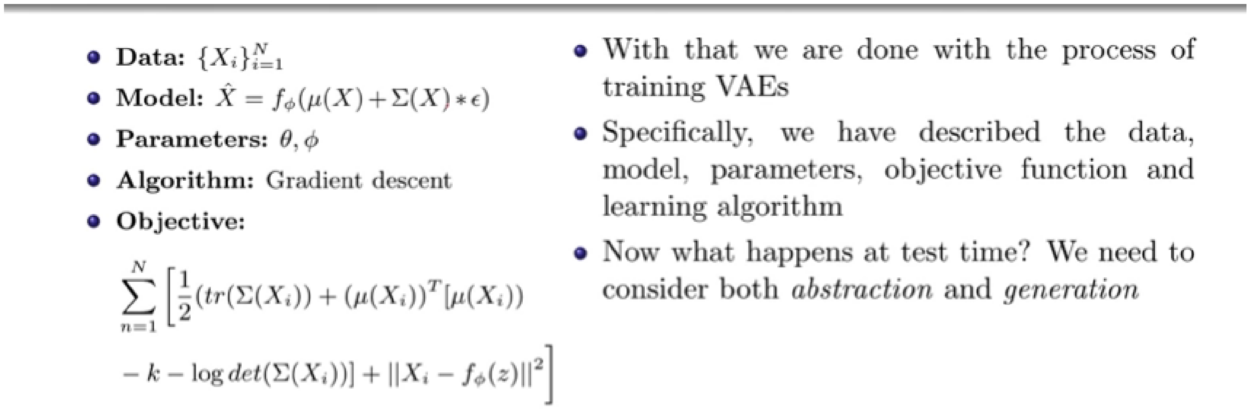
- VAEs use a neat trick to get around this problem
- This is known as the reparameterization trick wherein we move the process of sampling to an input layer
- For 1 dimensional case, given μ and σ we can sample from $\mathcal{N}(\mu, \sigma)$ by first sampling $\epsilon \sim \mathcal{N}(0, 1)$, and then computing

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

- The adjacent figure shows the difference between the original network and the reparamterized network
- The randomness in $f_\phi(z)$ is now associated with ϵ and not X or the parameters of the model

Now, varies use an eat trick, to get around this problem. And this trick is called the, it's called the, ‘Reparameterization Trick’. Where in what we do is, we move the process of sampling to an input layer, without me telling you anything else and assume that, I give you a method of achieving this English sentence, what does it actually mean? So here's, a hint. Right? we cannot get around the sampling, that was the whole premise of designing variation auto encoders, that we want to sample, we don’t want a deterministic function, but, we want the sampling to happen at a stage, so that it does not affect our part, you should still be able to get an, end to in part from the input to output, is that wish list. Okay? Is that, object to make sense, to sampling for all I care, but, just stay off my path. Right? Just don’t come in the path from, the input to the out. So, in that context can you read the sentence and try to see what would happen, it's not really obvious, I mean, but I just want you to think about it, so that, at least it opens up something in your brain and when I give you the solution, at least maybe you'll be able to relate to it. Right? Anyways we have time, so it doesn't matter about it. Right? So let's see, what I mean by that Right? So, before I get into the trick, let me just tell you something about normal distributions Right? I mean nothing profound all of you know this. But, suppose you have one normal distribution, which has parameters mu and Sigma. And another normal distribution, which is the standard normal distribution and as parameter 0 comma 1. Okay? Now, what I could do is? If I want to sample from the distribution mu comma Sigma, what I could do is I could sample from the distribution 0 comma 1? Because that’s easier, I can I easily find a library which can do that, and once I sample from that distribution, all I need to do is move the mean and it just for the variance. Right? So this term, because it's addition it makes sure that I move the mean and this term since its multiplicative, its make sure that I adjust for the variance, does that make sense? Right? So instead of variance 1, now the variance becomes Sigma, answer the mean 0 I have shifted it by mu, this is a standard trick, which probably are down in some of the courses that you have taken. Right? if not, at least one I’ve just smug it up and go back and look it up later on, but what it means is that, instead of drawing from you Sigma, I can draw from 0 1 and then just use the MU and Sigma to exist it. Now, based on this intuition, can you rethink about, what can be done? I have computed mu and Sigma, I don’t want to sample from mu and Sigma because if I do that, mu and Sigma depend on some parameters, the moment I sample from something which depends on parameters, my chain breaks. So I don’t want to sample from n mu comma Sigma, again I don't expect you to immediately, arrive at the solution but, do think about it, writing this thinking is what is important. The solution is of course there on the slide, that's why we have three more slides. Okay? So let's see, so what I'm going to do is? I'm going to take the X, I'm going to compute mu and Sigma till this point, everything is deterministic, there's no problem. Okay? Now, I’m not going to sample from the distribution mu and Sigma. What I'm going to do is? I'm assuming that, I have another input which is epsilon. Okay? I’m going to sample from that, so I can think of it that I had the input X,, in addition I had an input epsilon, which came from a normal distribution. Right? And now, whatever mu and Sigma I have drawn, I will just add, I will just do this operation. Right? This is exactly what I am doing, I am multiplying the epsilon by the variance and I'm adding the MU, I am getting a different quantity. Okay? Now, in terms of the parameters of the network, it is a deterministic, is it a deterministic function of the input, I’ve pushed all the randomness to Epsilon. Right? Do you get that? How many if you get this? Please raise your hands. If you don't can you write the model equation now and see if you can write it as an, end-to-end function of the input, the output has an end-to-end function of the input, with an input now is X plus some Epsilon, I don't care about what happens here. Because, I'm not going to back propagate anything to the epsilon, I don’t care about that. I only care about these parts remaining clean, does everyone get this, can I move on, please raise your hands. If you get this. Okay?

Refer Slide Time :(35:40)



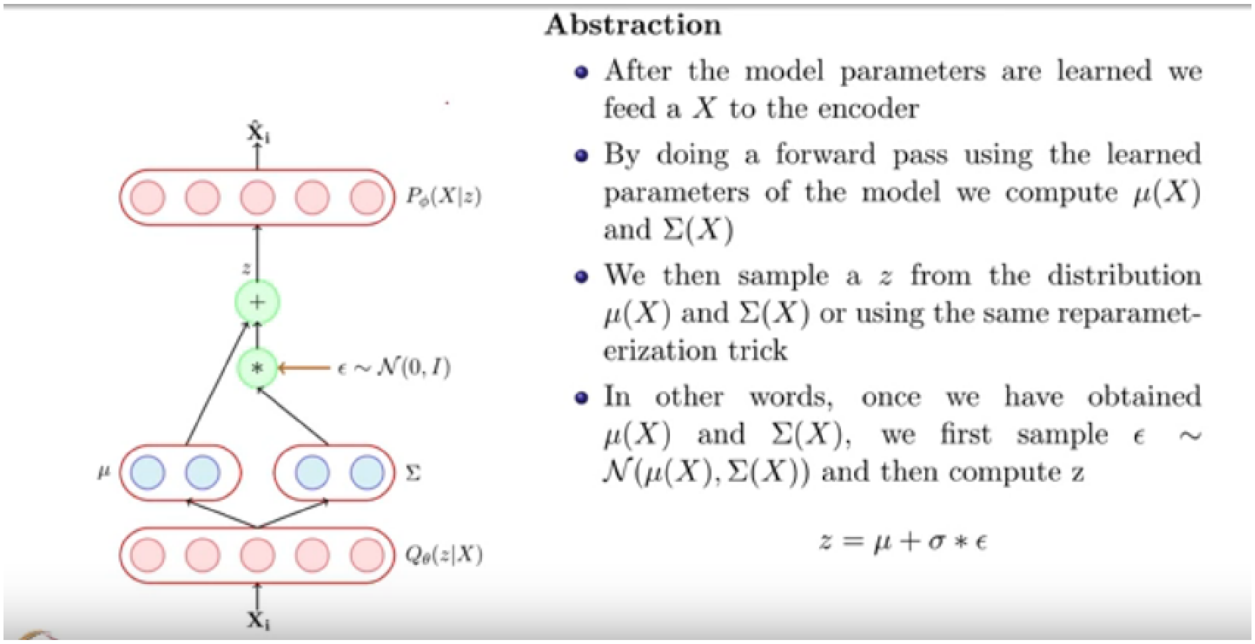
- **Data:** $\{X_i\}_{i=1}^N$
- **Model:** $\hat{X} = f_\phi(\mu(X) + \Sigma(X) * \epsilon)$
- **Parameters:** θ, ϕ
- **Algorithm:** Gradient descent
- **Objective:**

$$\sum_{n=1}^{N}\left[\frac{1}{2}(tr(\Sigma(X_i)) + (\mu(X_i))^T[\mu(X_i)) - k - \log det(\Sigma(X_i))] + ||X_i - f_\phi(z)||^2\right]$$

- With that we are done with the process of training VAEs
- Specifically, we have described the data, model, parameters, objective function and learning algorithm
- Now what happens at test time? We need to consider both *abstraction* and *generation*

So, with that we are done with the process of training variation Autoencoders. We have described the data, the model, the parameters, the objective function and a learning algorithm and here's the modeled. I have mu and Sigma, which are functions of X, parameterize by theta, there is no randomness here, the randomness has been moved to epsilon. Right? So still in terms of the input, my output is a deterministic function of the input and the randomness has gone to Epsilon, everyone sees this now, this is very, very crucial is one of the neatest tricks, so please make sure that you understand this. But, does there's no point in Marius not encoded, because without this trick you cannot train them, so it's very important that you understand this, avian fine with this. Okay? So, now we can write everything once you can write the model, that means the output as an end-to-end function of the input, then you are done. Right? Now, you can back propagate. So, now everything falls into place. Now, what happens attest time? What do we need to do at testing? What are the two things that we are interested in? Abstraction and Generation, remember. Vanish when we started this lecture, I said that in an auto encoder, you take an X and you leave reconstruct an X, what's the fun in this? Where’s the magic trick here? Right? You took an X, you just got an X, back so where’s the magic here, there is no magic here. Okay? So we will get to that answer now. But, what once you have trained the model, once you have done all this epic saga, what do we want to do now? You want to either use this model for abstraction or if you want to do for generation. Right? So, let's look at the abstraction first, what does that selection mean? That I will give you an input and you will computer, hidden representation that means, I will give you an X and you're going to compute the Z. So which part of the network is going to kick in? Encoder/decoder both, nothing, everyone, 630 but everyone encoder. Okay? Good.

Refer Slide Time :(37:45)



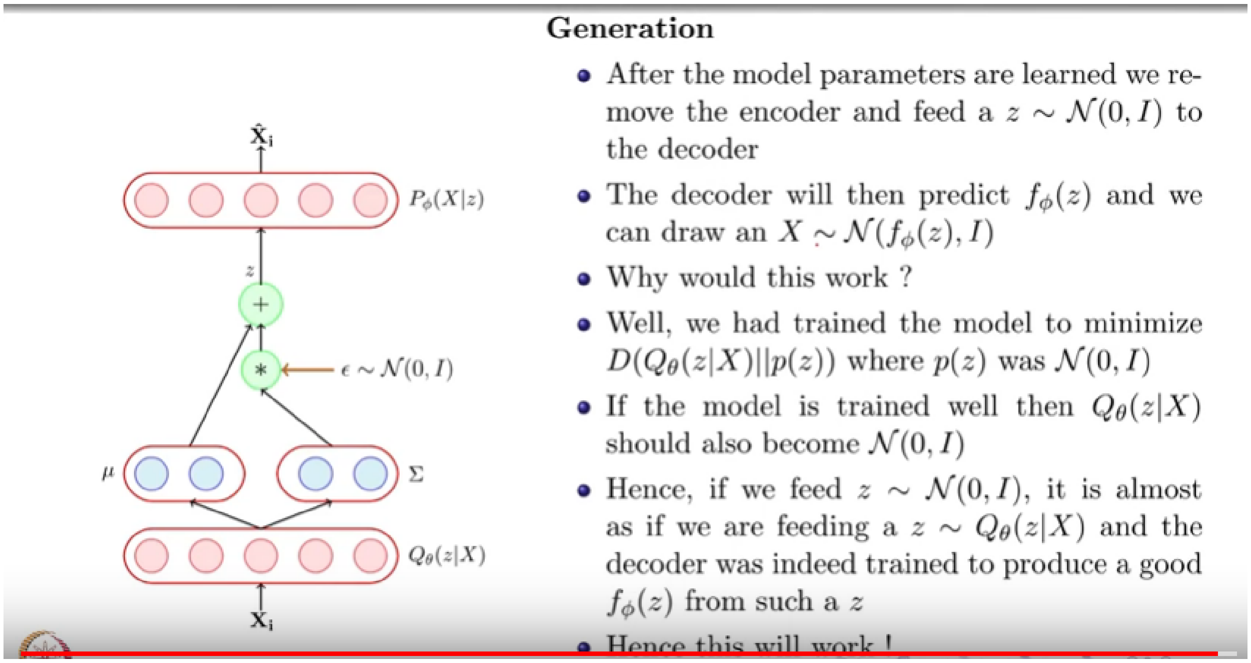
Abstraction

- After the model parameters are learned we feed a X to the encoder
- By doing a forward pass using the learned parameters of the model we compute $\mu(X)$ and $\Sigma(X)$
- We then sample a z from the distribution $\mu(X)$ and $\Sigma(X)$ or using the same reparameterization trick
- In other words, once we have obtained $\mu(X)$ and $\Sigma(X)$, we first sample $\epsilon \sim \mathcal{N}(\mu(X), \Sigma(X))$ and then compute z

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

So force for abstraction, I'll give you X, what with the encoded predict? It will give me a Z. Right? The output of the encoder is going to be a Z. Right? I give it an X; the output of the encoder is going to be a Zed. What does the encoder give me? Mu and Sigma. It gives me a parameters of the distribution Zed given X, it does not give me Z .what do I want in abstraction? Z. So now, there’s this gap. Right? The encoder is not capable of giving me my Z, what will I do? I will sample from that. Right? So the encoder will give me mu and Sigma, I’ll sample from that and I will get is Z. So now, unlike an auto encoder, which for a given X always gives me the same Z, what will happen in a variation auto encoder? I could get different Z’s. Right? It’s each time a sample, I should get a different set, of course in proportional to the parameters of the distribution, but I will still get this different Z, it's no longer deterministic. Okay? and how you sample is up to you, you could either sample from the distribution, mu comma Sigma or you could again use the same parameterization trick, you could sample from the distribution 0 comma I and then do this shift of variance and shift of mean. Right? But, the bottom line is the encoder does not produce a Z, it produces a distribution or rather, it produces the parameters of a distribution and then, you can sample from that, distribution. Okay? So that’s what? This slide essentially says, so a question is that, mu and Sigma were trained in a way that they become close to 0 comma I? Right? that means, my latent distributions, my latent variables come from the distribution 0 comma I. so, I kind of just sample from that distribution, why do I even need to feed an X? I don't even care about the X? I just sample from the normal distribution and say this is the abstract representation. So, I'm going to give you an answer and then I'm going to contradict that answer, when I talk about generation and that's what happens in optimization problems, there's always this trade-off. Right? So, that's our objective, that the KL divergence will be minimized, will it actually collapse to 0comma I, for all the x's, that's not clear. Right? So what will happen in practice is? You learn these distributions, which are all means shifted and variants shifted, because you have not been able to drive the mean to zero or the variance to unity. Right? It will still, the means would still be different for these X's, they'll be as close to the normal distribution as possible, but stills lightly different. That’s why you need to feed in this X to get that particular distribution and sample from them. Okay? I’m going to contradict this story, when I talk about generation, but everyone gets this. Okay? Fine.

Refer Slide Time :(40:37)



Generation

- After the model parameters are learned we remove the encoder and feed a $z \sim \mathcal{N}(0, I)$ to the decoder
- The decoder will then predict $f_\phi(z)$ and we can draw an $X \sim \mathcal{N}(f_\phi(z), I)$
- Why would this work ?
- Well, we had trained the model to minimize $D(Q_\theta(z|X)||p(z))$ where $p(z)$ was $\mathcal{N}(0, I)$
- If the model is trained well then $Q_\theta(z|X)$ should also become $\mathcal{N}(0, I)$
- Hence, if we feed $z \sim \mathcal{N}(0, I)$, it is almost as if we are feeding a $z \sim Q_\theta(z|X)$ and the decoder was indeed trained to produce a good $f_\phi(z)$ from such a z
- Hence this will work !

So now, for generation what are we interested in doing? Given X, reconstruct the X. that's what generation is right? I take an image, reconstruct it and give it back to you Right? And pass it through my GPUs, will have many of those and then give it back to you. What’s the generation problem? From thin air. Right? I'll give you a Z and you have to give me back an X. How do we do this generation? Auto encoders could not do this because; we did not know what z to give it. Now, what will we do? Sample from their? Sample from which distribution, standard normal distribution why? Why do we sample from the standard normal distribution? What kind of Z's were we interested in once which were likely given X. Right? those were the kinds of Z's, that we were interested in, not any Z from that entire large high dimensional space, we were interested in Z given X. what do the training ensure? The distribution of z given x is goes close to the standard normal distribution. So, now and that’s where my contradiction is Right? Now, at test time, if I draw from the standard normal distribution, I know it's as close to; some of the Z's that I had, said during training. does that make sense, what I'd done during training is, I had made sure that the distribution Z comma, Z given X, goes as close to the standard normal distribution as possible. The answer which I gave to her as, I said that, it’s close, but not very close, but I can still deal with it, it's still going to be close to the standard normal distribution. So, now if I sample from it, it’s much better than, sampling in the blind from the entire high dimensional space, it's still closer to the distributions that I have learned, all of them are closer to this standard normal distribution. Hence, if I draw from this distribution, I am feeding the decoder something, that it can deal with, because it comes from a distribution that it had seen during training day, does that make sense. How many of you understand that please raise your hands? Okay? Good. so that’s what I'm going to do, at test time, I’m going to draw from the distribution 0 comma I. the decoder will take this and it will predict the mean and then, I'm going to sample an X, from this mean. So again the decoder is not deterministic, it does not take a Z and give me a fixed X, from the same latent variable, I can generate multiple images, multiple sunny beaches with white sand and no people. Right? I could have multiple, images with the same latent configuration, does that make sense. So, that's what I'll do at, the time of generation and the rest of the slide just explains why this works because we have trained it to be close to the standard normal distribution .okay? So, sing, think about these, these two questions are important Right? So, on one hand, her question that, if it's all becomes standard normal distribution, why there bother about feeding it in X? You could just say that, don’t show me or X, here the latent distribution which comes, later in representation which comes from the standard normal distribution, it’s fine. Right? But, the answer to that which I gave was that, it's not going to be, it’s not going to collapse on the standard normal distribution, it's just going to be close to it and it would be differently close to it, depending on the value of x. hence, you need to compute the MU and Sigma given the X. Right? But, however at the time of generation, I am saying that, all of these are close to each other, so the approximation just draw from the standard normal distribution. Right? So this, I'm just doing a tradeoff between the goals at abstraction and generation. Okay? Is that fine. Okay? Someone actually showed me a thumbs up. Okay? Good. For first time, okay. So, we are done with variation Autoencoders. Now, we have three more lectures: Tuesday, Wednesday and Thursday. Thursday 8:00 o'clock, we have a lecture on Thursday. Right? Yeah. So anyway, I think I need one lecture for auto regressive models and then one lecture for Ganz and perhaps 15 minutes to bring this entire, generative modeling story to an end. Right? Like, so we started with RBMs, VAE’s is autoregressive Morrison and Ganz. What are the similarities, differences between them? And what are what are the limitations or advantage of one over the other? Right? So, we already hinted at something that VAE’s in the end try to make similar kind of assumptions that, we made for RBM’s, not exactly the same, but, in the same taste and so on. Right? So, we'll try to close that. Hopefully I should be done in two lectures, if not, I will take the lecture on Thursday also. Okay. Thank you.