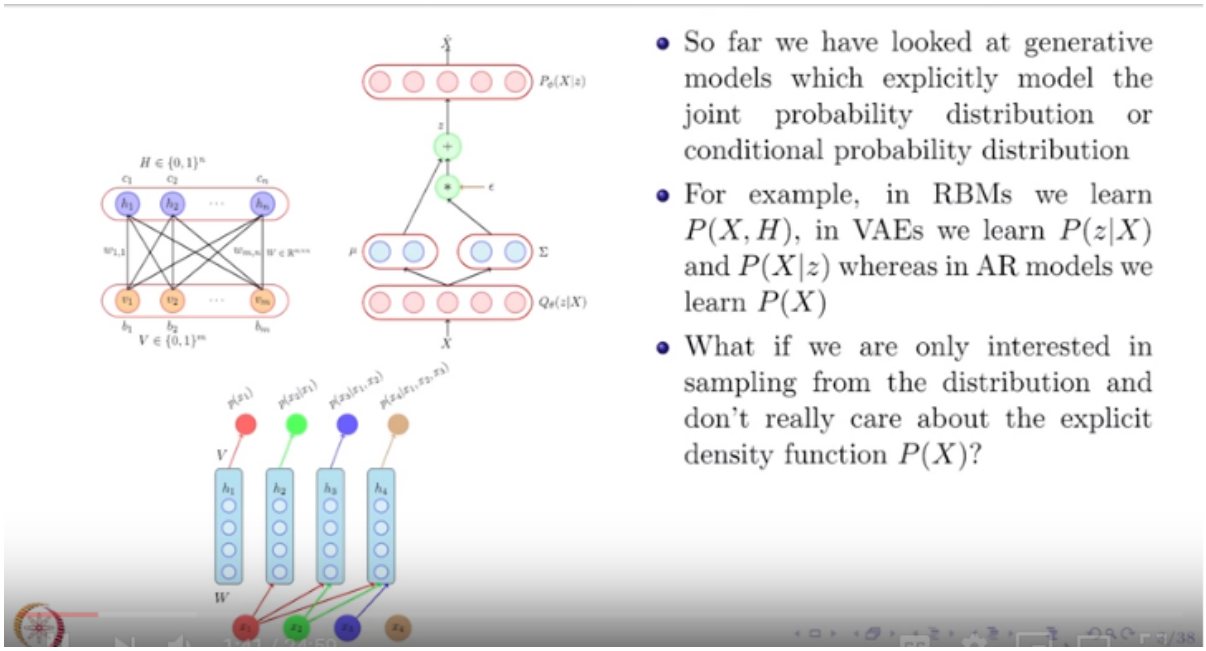
Lecture – 22.1
Generative Adversarial Network – The Intuition

Today we'll talk about generative adversarial networks, are popularly known as, ‘Ganz’. So, this is yet another family of models in the deep generative story. So, let's start looking at it. So, let's look at the intuition behind Ganz.

Refer Slide Time (0:35)

So, So, far we have looked at generative models which explicitly model the joint distribution or the conditional probability distribution. Right?

Refer Slide Time (0:39)



So, for example in the case of our, we learned P of X comma H in the case of VAEs we learned this P of Z given X and P of X given Z and in the case of autoregressive models we learned this P of X without any latent variables. Right? So, we were actually explicitly learning these probability distributions and then everything else was on top of red. Right? So, whether abstraction then Okay? We sample from P of Z given X if you want to generate we've sample from P of X given Z and So, on it's a both abstraction and generation were happening based on these explicitly computed probability distribution Okay? and each of them had their own way of dealing with it which like for example

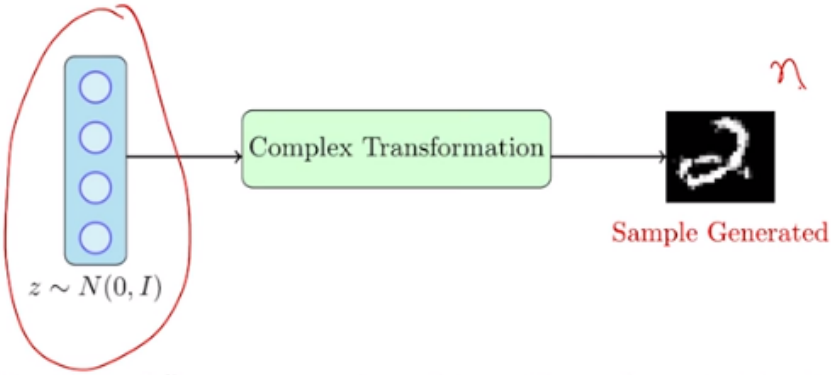
we're using sampling or variational inference, or using neural networks to parameterize the explicit factorization. Okay? Now, but, if what if you are only interested in generating samples, from the distribution and we don't care about what P of X actually is. So, let's try to understand this I've given you a lot of data let's say like giving you a lot of feminist images. What I'm saying is,

Refer Slide Time (1:43)

- As usual we are given some training data (say, MNIST images) which obviously comes from some underlying distribution
- Our goal is to generate more images from this distribution (*i.e.*, create images which look similar to the images from the training data)
- In other words, we want to sample from a complex high dimensional distribution which is intractable (recall RBMs, VAEs and AR models deal with this intractability in their own way)

that I've given you a lot of feminist images. I'm saying it, that I don't care what of P of X is or whether there's a latent variable, is a latent distribution and so, I don't care about all these things. I only care about the following, that you take all these images and now, give me more images which look like as if they had come from the same distribution. So, I've given you a lot of ways of writing1 2 3 4 and so on. Now, look at all this and give me more images which look like this data. I am not worried about what's the probability of this and so on. I just want to get samples from that. So, I am NOT interested in dealing with this P of X. Right? So, the goal here is to sample from this very complex high dimensional distribution and the operating phrase here is complex high dimensional distribution Right? and we know, that because of that things become intractable and we saw that RBMs variation autoencoders and autoregressive models had their own way of dealing with this intractability Okay?

Refer Slide Time (2:42)



- GANs take a different approach to this problem where the idea is to sample from a simple tractable distribution (say, $z \sim N(0, I)$) and then learn a complex transformation from this to the training distribution

But now what Ganz do is they completely bypass this whole process of learning P of X. There is no explicit P of X which is learned in Ganz. The idea there is simple you want me to generate images which look like our training images this is what I will do. I know it's very hard to sample from this complex high dimensional distribution. I know that I can easily sample from our normal distribution I'll take some Z belonging to some Rd I can sample from it and, then what I will do is I'll learn to make a very complex transformation starting from this Z. So, that I start producing images, which look like as if they had come from the training data. You get that? Did we see something similar earlier, taking a Z which is normally distribution and going to any kind of distribution, Variation auto-encoders. Right? There but, the difference was that we were starting from a D-dimensional Z and again going to a D dimensional different Z. Right? But now, they are even changing the dimensions because this would be D and this would be n dimension. Right? So, the image would be one zero two four. But, the latent variable, that he'll start with would be of 100 dimensions. But the idea is same that you could start from a normally distribution variable and learn this very complex transformation. So, that you start producing images. Okay? and the moment I talk about a complex transformation, what is the first thing that comes to your mind? please last lecture of the course? In your network. Okay? Good.

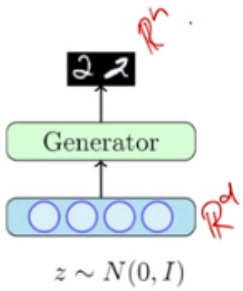
Refer Slide Time (4:07)

- GANs take a different approach to this problem where the idea is to sample from a simple tractable distribution (say, $z \sim N(0, I)$) and then learn a complex transformation from this to the training distribution
- In other words, we will take a $z \sim N(0, I)$, learn to make a series of complex transformations on it so that the output looks as if it came from our training distribution

So, we will take a Z normally distribution and try to learn this transformation.

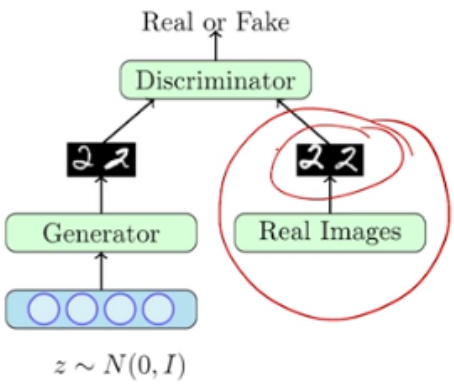
Refer Slide Time (4:12)

- What can we use for such a complex transformation? A Neural Network
- How do you train such a neural network? Using a two player game
- There are two players in the game: a generator



So, what can be used for such a complex transformation a neural network Right?
How do we train such a neural network? Okay? We are going to use a two-player game. okay? But, in the of course back. So, there are 2 players in this game there is a generator the job of the generator is what I just said. It has to take Z which is normally distribution. It has to learn this will be a generator will be some neural network, which will take this Z belonging to Rd and give me an image belonging to Rn. It will generate these samples. How we will see it's not clear yet. Right?

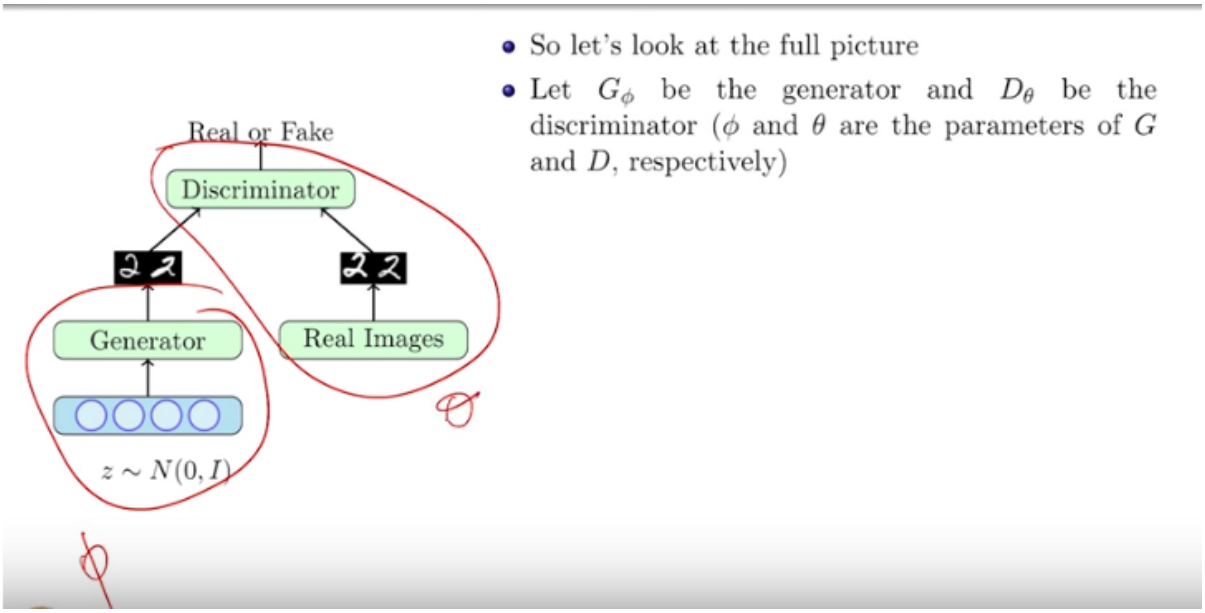
Refer Slide Time (4:48)



- What can we use for such a complex transformation? A Neural Network
- How do you train such a neural network? Using a two player game
- There are two players in the game: a generator and a discriminator
- The job of the generator is to produce images which look so natural that the discriminator thinks that the images came from the real data distribution
- The job of the discriminator is to get better and better at distinguishing between true images and generated (fake) images

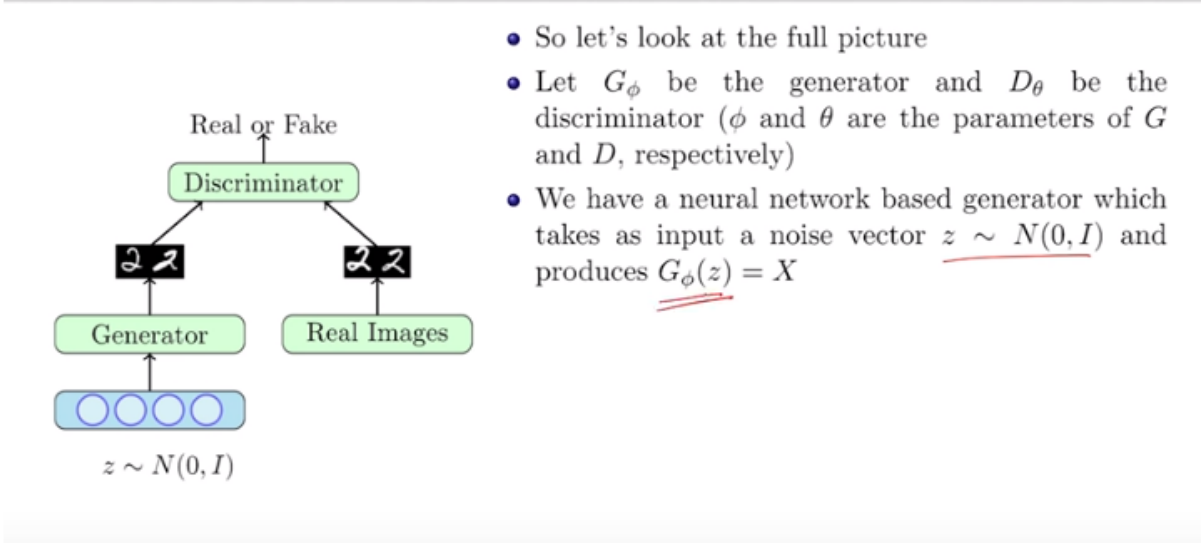
So, that is what the generator has to do, and then there is a discriminator. So, let's see how the generator and the disk committed interact. So, the job of the generator is to produce images, which looks. So, natural that the discriminating that the images actually came from the two data. Right? and the job of the discriminator is that it can take as input either images from your real training data. So, this is the actual Menes tata which I had given you. What I have circled? So, it can take images coming from the real Menes data or it can take images coming from the generator, and It has to become better and better at distinguishing between the generated images and the fake images. Right? So, you see this is being a two-player game each trying to kind of get the better of the other. So, the generator wants to generate better and better images. So, that a discriminating wish between them and the discriminator wants to get better and better at descript at distinguishing between images generated by the generator, and the true images. Do you get this set up? Does everyone get the set up? Please raise your hands. If you get right? Okay? Good. So, now, let's formalize this. Right? I mean this is all just intuition. So, let's just try to formalize it and, what I mean by formalizing is, that I have to answer the question how are you going to learn this right? So, what is the first thing, that I need to tell you the objective function. Right?

Refer Slide Time (6:02)



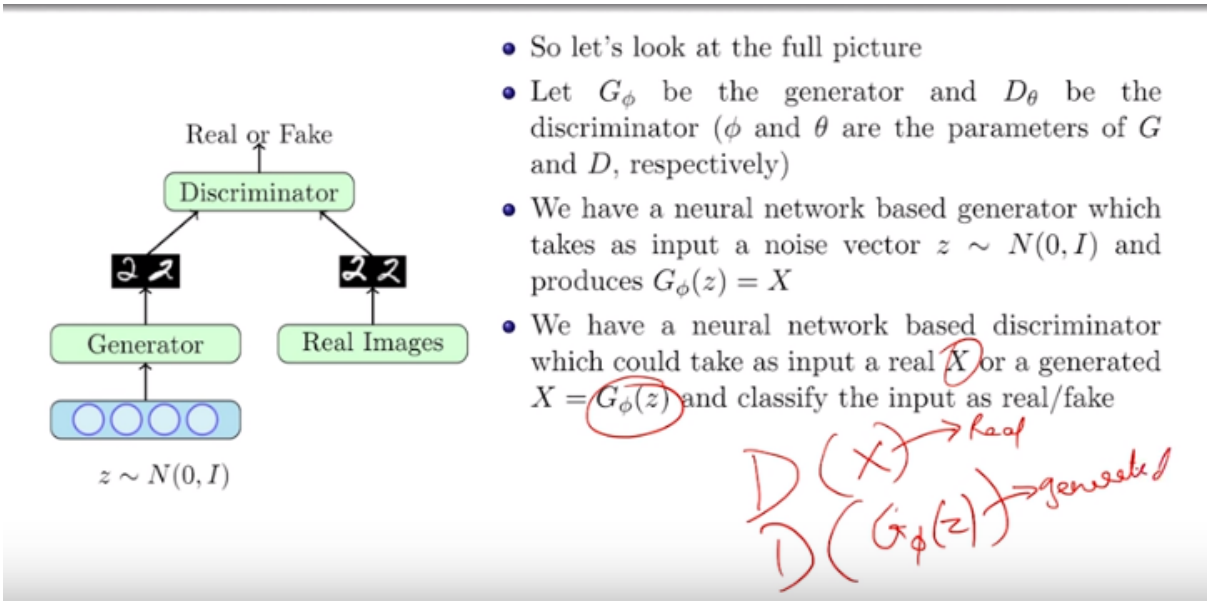
So, let's look at the full picture first. So, we have the generator, which is parametrized by five. So, these are the parameters of the generator this could be a feed-forward neural network or a conversion your network or. Right? But it will be a convolutional neural network, and you will have the discriminator which is parametrized by theta and this will also be a convolutional neural network in practice. Okay?

Refer Slide Time (6:26)



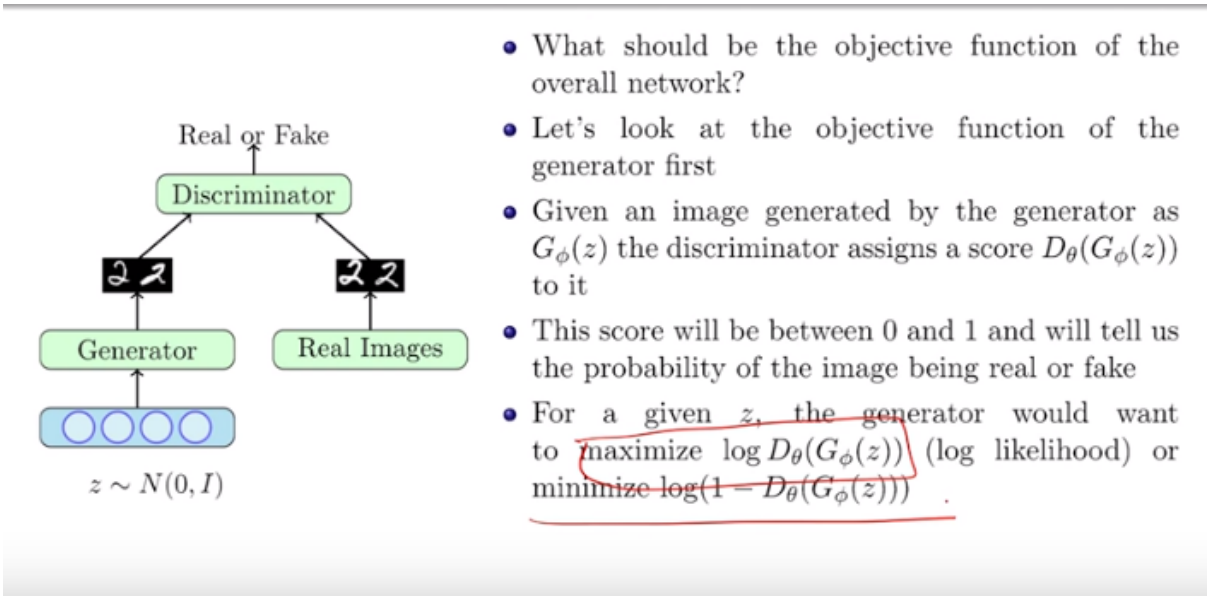
The neural network based generator is going to take Z as input and give us G Phi of Z. So, G Phi is the general function that I am using to represent the generator it will take Z as an input and give me an X,

Refer Slide Time (6:37)



and the discriminator is going to take an X the X could either be the real X or the generated X, and give me a score between 0 and 1. So, the output of the generator of sorry of the output of the discriminator, would be either d of X or it will be d of G Phi of Z, that means this is the real image this is the generated image, and this be the output D would be something in the range of 0 to 1. Right?

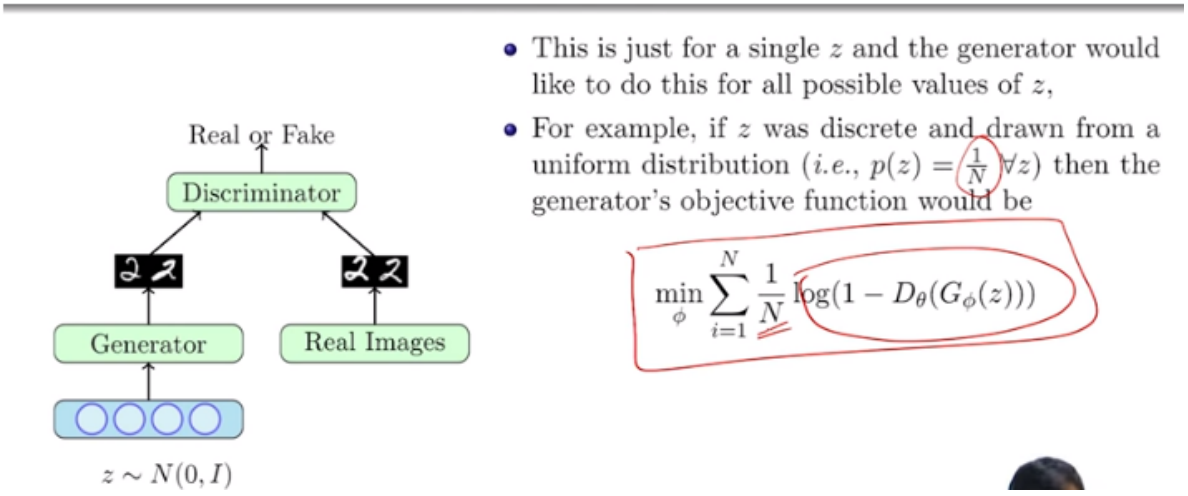
Refer Slide Time (7:04)



So, this D is going to tell us what's the probability of this being a real image, and what's the priority of this being a fake image, and by fake, I mean a generated image. Okay? Is that clear? Okay? Now, what should be the objective function of the overall network. Okay? at this point you get that the discriminator and the generator have different objectives or contradicting objectives, once need one needs to maximize something, the other needs to minimize something. Okay? If that is fine, I'm a tea that an intuition level if you get that then we will get into the details Right? So, we'll start with the

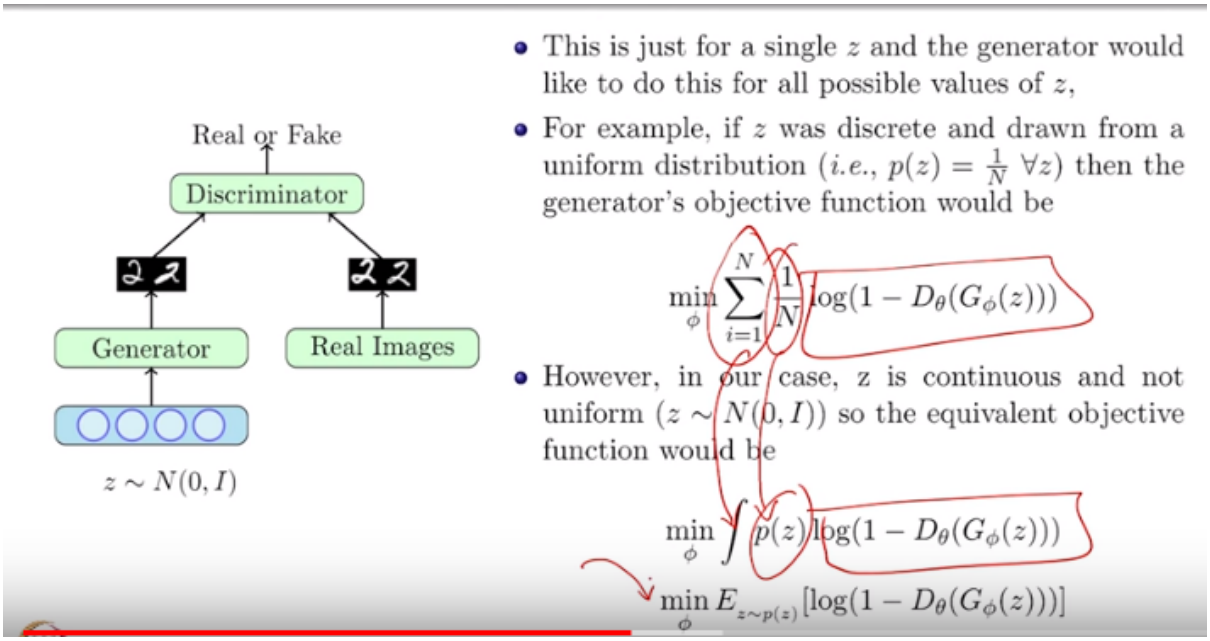
objective function of the generator first. Okay? So, given an image sorry you are given an image generated by the generator the discriminator, is going to assign a score to it. Right? So, this B is the score assigned to the discriminator to this generated image, and this score is going to be a bit number between Zero to one okay? That means the output is going to be a sigmoid Okay. So, the discriminator is going to assign a score between Zero to one, and it tells us where the image is fake or real. Now, what should the generator wish for. This course should be as high as possible can you write it as an objective function maximize, maximize log of D of G phi z. G theta D theta G phi z. Okay? Does that make sense? This is just a probability. Right? This is the log likely hood of the image being a real image Right? So, I can just maximize, that my object it could be to maximize, the score assigned, by the discriminator to this image. So, I am becoming very good at my job. I am being able to make the discriminator assign very high scores, to the images generated by me. and that that's the same as minimize the reverse of that. Right? So, D is a probability. So, instead of maximizing D. I am going to minimize 1minus D these are two equivalent objectives. Does that make sense Okay? Is this the entire objective function do you expect a summation or anything here? This is the objective function for one sample. But the sample does not appear here, which is the variable which appears here. this is the variable which appears here Z that's the input. So, this is the loss function for one, given input. Okay? So, what's the total loss function going to be? sum over. So, I like the answer which say sum over. What's the sum over going to be? Generator is a deterministic function. If once you give us, then it will give you the same image every time. Right? you can sample different sets. But one you fix a Z it will give you the rest of the neural network over the deterministic function. So, what I'm trying to impress upon, or what I want you to think about is, when we look at all the other networks, that we have done feed-forward neural networks, convolutional neural networks, or recurrent neural networks, the input was this X I I equal to one to N and then we should sum up the loss over all these X's. But now the input is this Z, and how many seeds do I have? As many as, I want Right? I have the entire set space. Because that's just how many times I can sample from ten zero one. Okay?

Refer Slide Time (10:11)



So, let's see what the objective function should be. So, what I showed you was the objective function for a single Z the generator should actually try to maximize or minimize whichever objective function you want to work with will work with the minimize one. So, the generator should try to minimize that quantity 1 minus D for all possible sets. Is it fine! Is that okay? Because, every Z coming from the normal distribution is a valid input to the generator all of these are valid inputs. So, it has to minimize this for all possible Zs. Okay? Now, suppose Z was discrete, and it came from a uniform distribution. Okay? Then the probability of any given Z would have been 1 by n. Where n is the total number of Zs possible, and in that case, you could have written this which you are very familiar with summation over all possible Zs. The loss function, and then take the average, which is 1 by n. Everyone is fine with this. If Z was disk read this is what you would have done and this is exactly what you do when you are given these capital n training examples. You take the loss function for one training example summit for all the training examples, and divided by n. So, this is very similar to that. Is that fine? But, Is this okay for us? Yes or no? why? Z is first of all it's not discrete. Second is it does not come from uniform distribution it comes from a normal distribution. So, can you tell me, what is the modification, that I need here? Summation will get replaced by, integral, and one by n will be replaced by P of Z. Good.

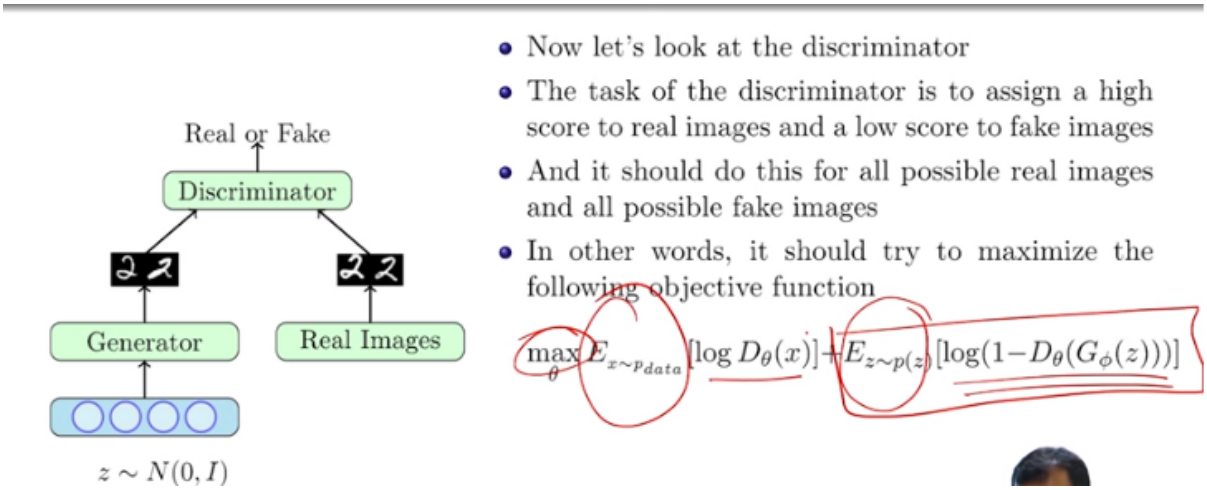
Refer Slide Time (11:57)



So, that's exactly what I've done. Right? So, this remains the same 1 by n which was the uniform probability, gets replaced by the normal probability. Okay? and summation gets replaced by the integral. Is that fine? and this you can actually write as the expectation, of the quantity in the box. This is of the form P of X function of X. So, you can write it as the expected value of the function of X under the distribution P of X. Is that fine okay? Of course, instead of X we have Z's here. So, this is

what we are going to minimize. So, remember that the goal of the generator, is to minimize this expected loss, over all possible values of Z, and it's a minimize objective function the objective function is to minimize the quantity, that is you see in the bracket. Okay? So, that's for the generator.

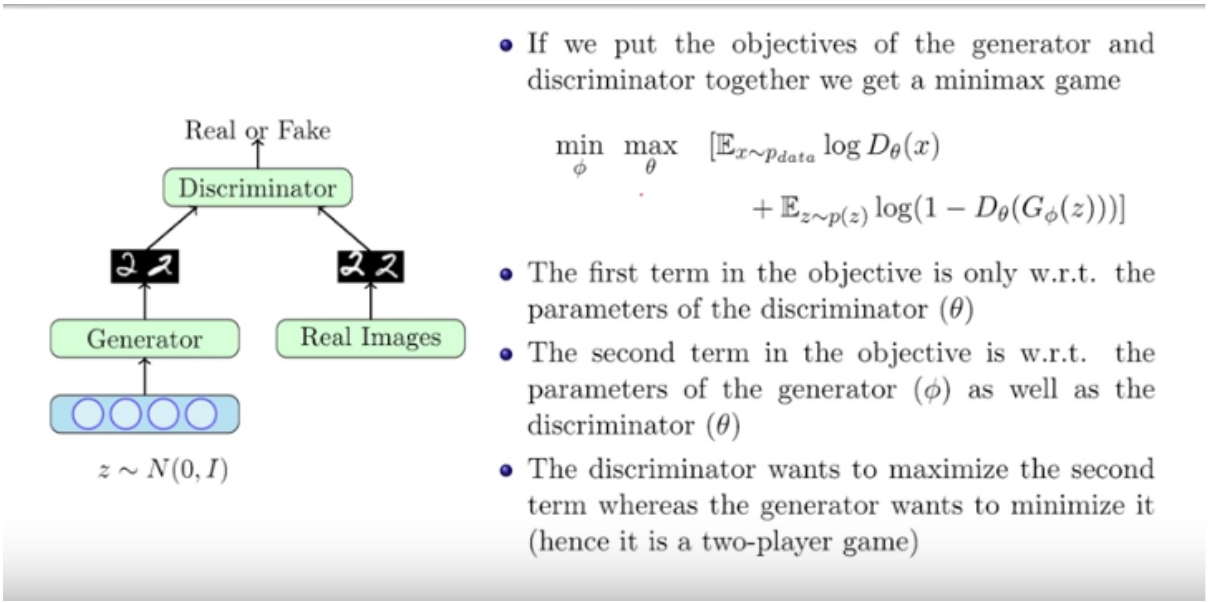
Refer Slide Time (12:47)



Now, let's look at the discriminator. What should a discriminator do? The task of the discriminator is actually two fold. It has to assign a high score, to real images, and low score to generated images. Right? In the case of generator, it was only onefold that means it has to make sure that the image is generated by it or given high scores. So, you had one expectation there. Now, from the discriminator, I have two expectations. So, the loss function will also, have two terms and two what will these two terms be expectations. kill the joke. Okay? These two terms will be two expectations. Right? What would these expectations we over? For every possible Z, what do I want to do? Minimize the score assigned to the generated image. Right? That's the same as maximize score one minus that score. Is that fine? So, the expectation over Z is going to be log of one minus DZ. What about the other guy? For the samples coming from the true distribution what do I want? Maximize the score. Okay? Is this fine? This is for all possible Z's. I want to maximize one minus the score. Right? That's the same as, minimizing the score. Is that okay? Everyone gets this? and then for all possible X's coming from the true data. I want to maximize this goal So, the reason I am writing it as one - because then I can write it as a single maximize objective. Right? So, maximize the score assigned to the true images, and one minus score assigned to the fake images. everyone understands these two expectations please use your hands. If this is clear okay? and you also, agree, that these two expectations actually boil down to two integrals. Right? Because both X and Z are continuous. Okay? Fine! So, see this was maximize you had maximization of this, in the case of the generator, what did you have? Minimizing the same quantity. Okay? So, now can you tell me what's the overall X is continuous here? So, you're saying you could approximate by the data that you have? Yeah, So, we will come to that Okay? Is that fine? Now, given I've told you what's the generator expectation? I mean generators loss function, and I have told you what's the discriminate as loss function. Can you put these together, and give me the overall

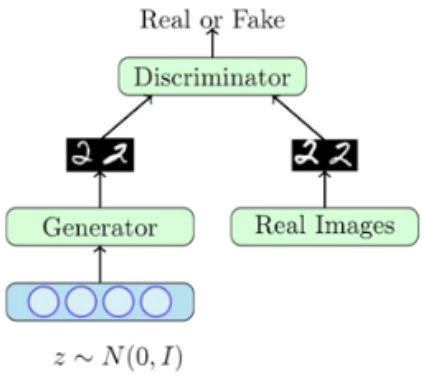
loss function? Okay? What would be the whatever is that objective function? What would be the object and what would be the optimization with respect to what are the parameters of the objective function? Theta in Phi the parameters of the generator and the discriminator. Okay? Now, I already told you, that this is a two-player game someone tries to minimize something, the other guys tries to maximize something. Putting all this together, that means that the parameters R Theta Phi. One guy wants to minimize the other guys guys wants to maximize and you've also, seen what one guy wants to minimize, and you have seen what the other guy wants to maximize can you put all this together and give me one single objective function. It's not very obvious. But if I give you the answer it would be very obvious. But the reason I'm asking you is, that I want you to think about it. Min max off. Okay?

Refer Slide Time (16:20)



So, this was the overall objective will look like, right? with respect to theta you want to maximize this quantity, with respect to Phi you want to minimize this quantity. Okay? and remember, that the first term only depends on theta there is no Phi there. Okay? So, even if I want to minimize this with respect to Phi it does not matter, because when I will compute the gradients, that will go to zero. Its I can just put this term inside even though it does not depend on Phi, because when I compute the gradients with respect to Phi it will go to 0. Is that fine Okay? Yeah, no, because with respect to theta you want to maximize this objective function Okay. So, the second term depends on both Phi and theta. Fine! Now, the discriminator was just maximize the second term, and the generator wants to minimize the second term. Is this clear? So, far everyone gets this it's fine Okay? Good I'm not seeing any blank faces today

Refer Slide Time (17:14)



- So the overall training proceeds by alternating between these two step
- **Step 1:** Gradient Ascent on Discriminator

$$\max_{\theta} \left[\mathbb{E}_{x \sim p_{data}} \log D_\theta(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_\theta(G_\phi(z)))\right]$$

- **Step 2:** Gradient Descent on Generator

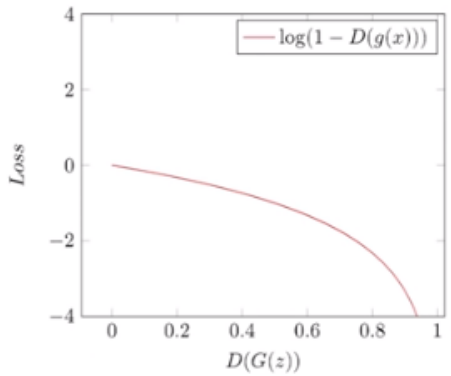
$$\min_{\phi} \mathbb{E}_{z \sim p(z)} \log(D_\theta(G_\phi(z)))$$

- In practice, the above generator objective does not work well and we use a slightly modified objective

So, the overall training will proceed by alternating between these two objectives. So, step one would be to do gradient ascent on the discriminator. Why I sent because you want to maximize. Okay? So, we'll take the gradient of this quantity with respect to theta. Okay? and then do a gradient ascent on that, and the step two would be to do a gradient descent, on the generator with respect to this objective. Is it fine? So, you're going to alternate between a maximize, and minimize problem. We'll do one step of maximization then one step of minimization and each of these the parameter with respect to which you're optimizing is going to change. Okay? Is that fine. Okay? Now, we still have this problem. So, in practice what happens is that this particular objective function, that you have chosen for the disk for the generator does not work very well, and we'll see why it does not work very well.

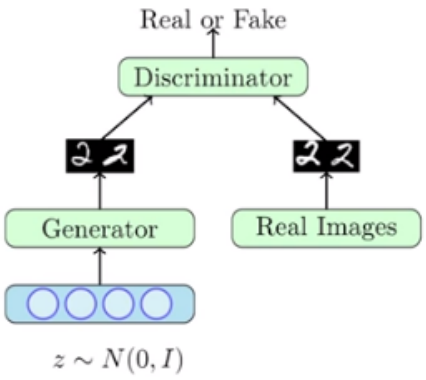
The only reason we wrote it as this remember that I could have actually written it as maximize this Right? Whilst I wrote it as minimize the opposite of it the reason, we did that is we could write it as a neat minimax game, then one guy is trying to minimize this the other guy is trying to maximize this. But actually, what you're interested in is maximizing this score. Okay?

Refer Slide Time (18:43)



- When the sample is likely fake, we want to give a feedback to the generator (using gradients)

Refer Slide Time (18:45)



- So the overall training proceeds by alternating between these two step
- **Step 1:** Gradient Ascent on Discriminator

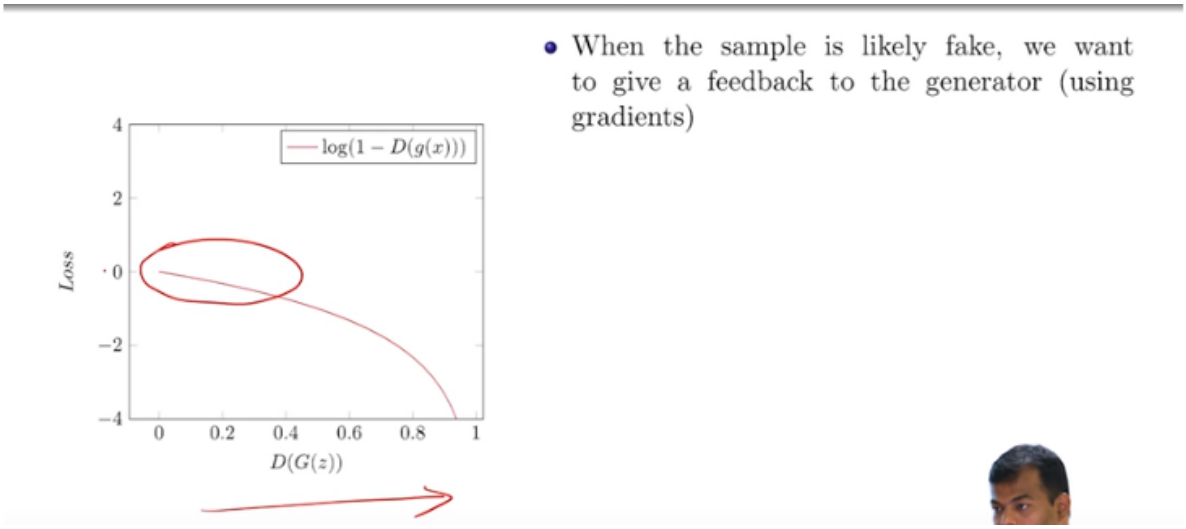
$$\max_{\theta} \left[\mathbb{E}_{x \sim p_{data}} \log D_\theta(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_\theta(G_\phi(z)))\right]$$

- **Step 2:** Gradient Descent on Generator

$$\min_{\phi} \mathbb{E}_{z \sim p(z)} \log(1 - D_\theta(G_\phi(z)))$$

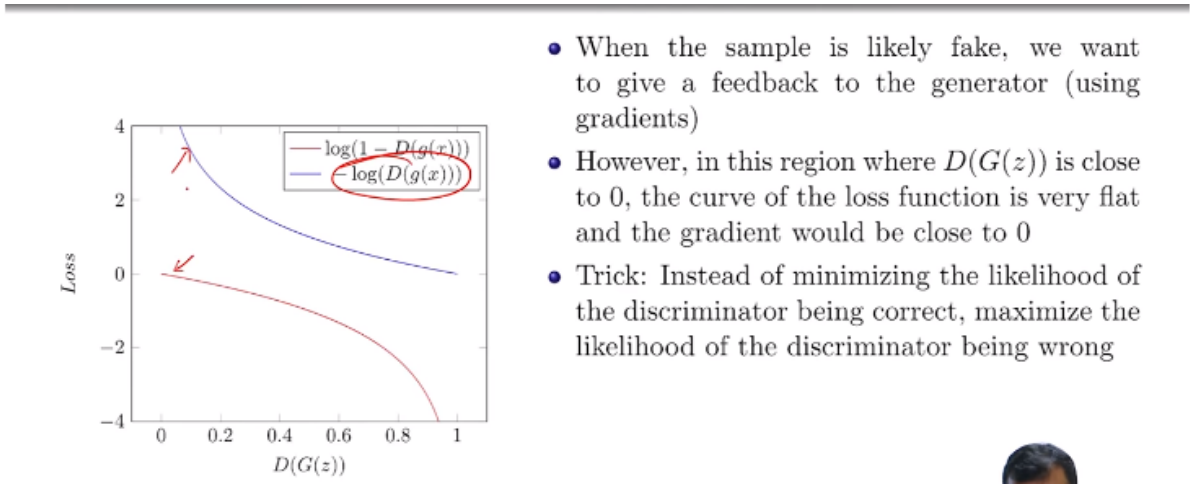
- In practice, the above generator objective does not work well and we use a slightly modified objective
- Let us see why

So, it turns out, that in practice if you use this objective, where you want to minimize one minus the score assign.
Refer Slide Time (18:51)



Then what happens is, that see this is the score assigned on the x-axis you have the score assigned by the discriminator. Okay? for fake images the score is going to be very close to zero. Okay? If that is the case what is the gradient going to be. So, see in this region actually, the slope is very flat. So, what is the gradient going to be, zero So, that means a generator will find it very hard to learn. Okay? Because imagine in the beginning of the training. Right? The generator is really producing fake images. Because it does not learn anything. So, it's really going to be very very low score assigned by the discriminator, and then that's why the gradients will not flow back. So, if you try to minimize, minimize this quantity, then it is going to be difficult.

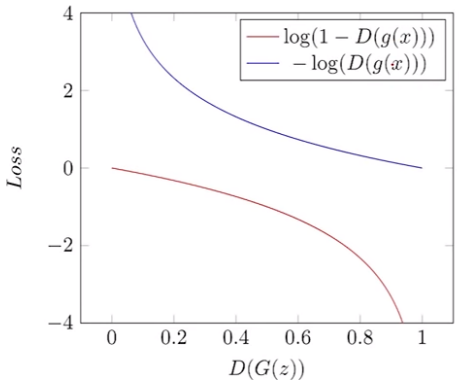
Refer Slide Time (19:39)



So, instead what we do is we maximize 1minus of this. Right? That means we maximize the log of D of GX. Now, that curve looks slightly different. Because it is 1 minus this. So, wherever this was small, actually this is going to be large. Do you get that? So, the objective function still remains the same. Minimize 1 minus score, is the same as maximize the score. But, in the second case the gradients are now better, and the gradients will flow better, because of it a generator can learn better. Does that make sense? Everyone gets that. Yeah, but, generally Right? The objective of the discriminator is much easier because just a classification. Right? So, the discriminator can learn much

faster the generator has a harder job it has to get all these 1 0 to 4 pixels Right? and the generator just has to predict one value. So, even if initially both are random the generator Soon starts learning very well. Right? Because it's much easier job for the generator. Right? Okay? now So, just to summarize Right? Okay? Now so just to summarize Right? we started by saying that the generator should behave in a way such that it maximizes the score assigned by the discriminator Okay? But then we converted that objective to minimize one minus the score and that was only so that I could have written that entire thing neatly as a minimax problem Okay? Then we come to the more practical issue that Okay? This objective looks fine in theory but you will have this problem that the gradients will not flow properly so that's why we go back to our original objective which was to maximize the score assigned by the discriminator and in that case the gradients are much better in this initial portion and the generator can learn better does that make sense everyone is fine with this please raise your hands if you are okay with this Okay?

Refer Slide Time: (21:23)



- When the sample is likely fake, we want to give a feedback to the generator (using gradients)
- However, in this region where $D(G(z))$ is close to 0, the curve of the loss function is very flat and the gradient would be close to 0
- Trick: Instead of minimizing the likelihood of the discriminator being correct, maximize the likelihood of the discriminator being wrong
- In effect, the objective remains the same but the gradient signal becomes better

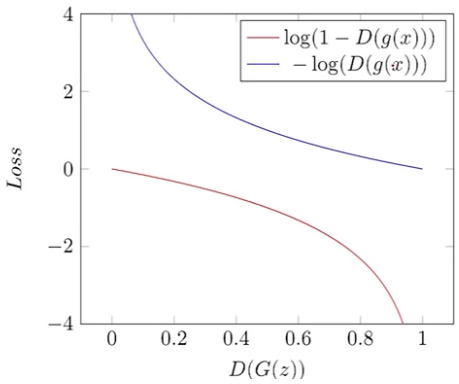
And of course?

Refer Slide Time: (21:23)

With that we are now ready to see the full algorithm for training GANs

1: **procedure** GAN TRAINING

Refer Slide Time: (21:24)



- When the sample is likely fake, we want to give a feedback to the generator (using gradients)
- However, in this region where $D(G(z))$ is close to 0, the curve of the loss function is very flat and the gradient would be close to 0
- Trick: Instead of minimizing the likelihood of the discriminator being correct, maximize the likelihood of the discriminator being wrong
- In effect, the objective remains the same but the gradient signal becomes better

in effect the objective remains the same right one minus minimizes the same as the maximizing guarantee Okay.

Refer Slide Time: (21:29)

With that we are now ready to see the full algorithm for training GANs

procedure GAN TRAINING
  for number of training iterations **do**
    for k steps **do**
      • Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, .., \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$
      • Sample minibatch of m examples $\{\mathbf{x}^{(1)}, .., \mathbf{x}^{(m)}\}$ from data generating distribution $p_{data}(\mathbf{x})$
      • Update the discriminator by ascending its stochastic gradient:

$$\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} \left[\log D_\theta \left(x^{(i)} \right) + \log \left(1 - D_\theta \left(G_\phi \left(z^{(i)} \right) \right) \right) \right]$$

    end for
    • Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, .., \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$
    • Update the generator by ascending its stochastic gradient

$$\nabla_\phi \frac{1}{m} \sum_{i=1}^{m} \left[\log \left(D_\theta \left(G_\phi \left(z^{(i)} \right) \right) \right) \right]$$

  end for
end procedure

So now we are ready to look at the full algorithm for training GANs and I think this is where your question will be answered so there will be some fixed number of training iterations. First for scale steps we are going to focus on the discriminator so what we are going to do is we'll take a mini batch. So remember that we had this expectation over z now we are going to approximate it using some M samples from Z so we are going to take some MZ. Okay? What are you going to do with the Z pass it through the generator you will get a G, G of Z what will you do with the G of Z pass it to the discriminator you will get a score what will you do with this core compute the loss function Right? and then back propagate Okay? so we have taken samples from the noise distribution will also take samples from the real distribution so remember for the disk you had expectations Z and expectation

X. So both these expectations we are going to approximate by samples, the X is easier because it's just from the training data and the Z we'll just take some Z's and just compute that loss function the expectation is going to be approximated by a summation is it fine. Okay? And now you're going to do a gradient ascent by computing the gradient of this with respect to theta and do a gradient ascent because you want to maximize is that fine does that make sense and the thing inside the bracket it just again the log likelihood rate we have taken the gradient of this a million times in the course. Right? So, this is again a very simple objective, and this is just a normal feed-forward neural network or a convolution network. So, once you cast it as this the rest of the stuff should be clear it again boils down to your favourite and only I'll go them which is back propagation Okay? what about the generator you'll sample many batch of M samples from Z Okay? and again the generators loss function was expectation with respect to Z so going to approximate that expectation using these samples and once you compute that loss right so you have X you have approximated the expectation using this empirical estimate you have the loss function you take the gradient of that I knew back propagate is that fine. Okay? So now you have the entire story ready you have the generator getting trained for K steps and then the discriminant getting trained for certain steps in practice now this so impact si training GANs is a bit tricky you have to do a lot of hyper parameter tuning get everything right and there are different papers which report some people say that k equal to 1 is fine. That means one step of discriminant step of generator but there are other papers which report different hyper parameters for this, Right? So training GANs, you really need to work with them a bit, to get them to work properly, but when they do, they generate very good images. Okay?