Deep Learning
Prof. Prabir Kumar Biswas
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture - 49
Layer, Instance, Group Normalization

Hello, welcome back to the NPTEL certification course on Deep Learning. So, last few classes we are talking about the normalization techniques and we have said that these normalization techniques takes care of the covariant shift which is a problem while training of the deep neural network, as with as the covariant shift is nothing but the change of the distribution of the data even if the data belongs to the same class, and as the classifiers they basically learned from the data distribution.

So, if the data belonging to the same class changes distribution from batch to batch. Basically the classifier gets confused that what to learn or what is the boundary between different classes. So, when you talk about this normalization techniques the normalization technique basically tries to reparametrize the data or repara parameterize the distribution with having some standard deviation say gamma and the mean beta where this gamma and beta they are tuned during the back propagation learning of the neural network and this ensures that even if there is a covariate shift of the training data from batch to batch.

But the covariate shift will be minimal or the shift of distribution from one batch to another batch will be minimal as a result the different layers of the neural network will learn independently of other layers, the training will be much more stable and the training will be much more faster. So, that is the advantage of I am employing normalization techniques along with your gradient descent procedure. So, that makes gradient descent procedure much more efficient much more fast.

So, the particular normalization techniques that we have discussed in your previous class is the batch normalization technique and we have also seen that what is the performance of batch normalization techniques that is with an example network. If you employ batch normalization (Refer Time: 02:51) and if you do not employ batch normalization, then we have seen that without a batch normalization the number of iterations required to reach say more than seventy percent of the validation accuracy. If you do not use batch

normalization is more than 30 million. In fact, it is almost 36 million iterations are required to reach and validation of accuracy more than seventy percent

And this was done this experiment was carried out on a particular architecture which is the inception architecture. And then we have seen, that when you use batch normalization. The number of iterations required to attain the similar performance of more than 70 percent of validation accuracy it is just equally reduced and. In fact, from 36 million from about thirty million it becomes less than one million or so.

So, this batch normalization techniques improves the training the learning rate much more faster. And you can train the neural network in very less number of iterations.

(Refer Slide Time: 04:20).

So, in today's lecture we will talk about the other normalization techniques like we will talk about layer normalization, we will talk about instance normalization and we will talk about group normalization. Now one of the problems with batch normalizations operation we will see later is that the performance highly depends on batch size.

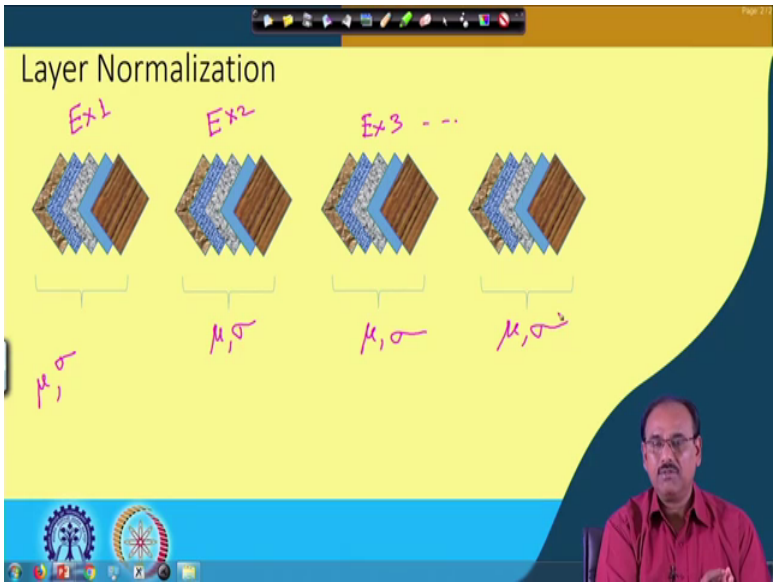
If you reduce the batch size then the performance degrades or as if the batch size is more the performance is quite acceptable. And the other disadvantage with batch normalization is that you cannot normalize until and unless the entire batch is operated on. Because when you compute the mean and standard deviation you are computing the mean and standard deviation over the number of channels, over the features in the same

channel computed by the same convolution kernel, but on different examples in the same batch.

So, as a result I cannot normalize the data until and unless the entire batch is processed. And that becomes a difficulty because in some other kind of network for example, in a recurrent neural network or rnn such batch normalization techniques cannot be employed. Similarly in gradient descent in instead of batch stochastic gradient descent if we simply use stochastic gradient descent, where the weight updation is done sample wise. With every training sample you compute the error, you compute the error gradient and accordingly you can normalize or you can update the weight vectors. There your operation has to be done sample by sample.

So, it cannot be done in batch normalization because in batch normalization you have to wait until and unless the inter batch of the training samples is operated on. So, these are the problems which are addressed in what is known as layer normalization and this can also be addressed in instance normalization.

(Refer Slide Time: 06:33).



So, let us see that what this and layer normalization is. So, as we have seen that in case of batch normalization when you go for normalization what you do is you take this training this feature from one batch, the identical feature from another batch, the identical feature macron and another batch and so on.

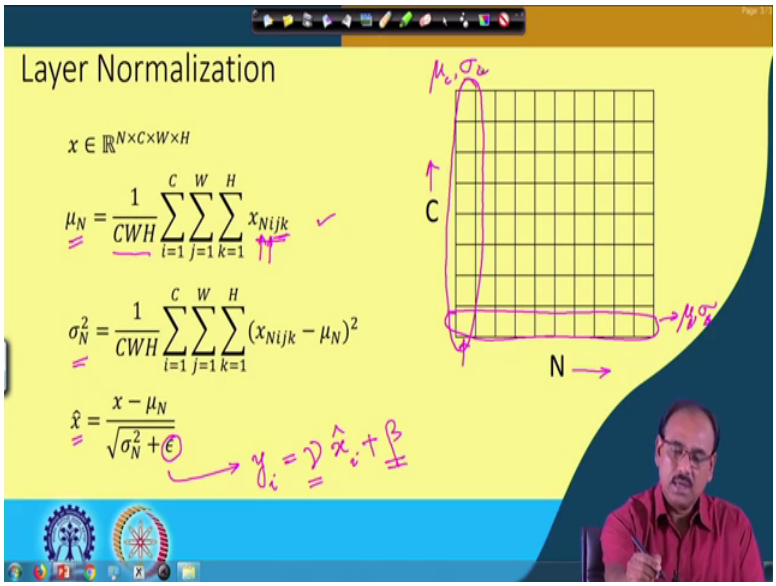
And then you compute the mean and standard deviation with respect to these feature maps. Whereas, in case of layer normalization what you do is you take the features of the feature maps computed by the different kernels convolution kernels over a single example. So, these are the feature maps that you get from say example 1 or training data 1, this you get from training example 2 this we are getting from training example 3 and so on.

And when you compute mean and standard deviation the mean and standard deviation are computed over these features feature maps computed from example 1. Similarly these feature maps computed from example 2 another mean and standard deviation will be computed here, another mean and standard deviation will be computed here, another mean and standard deviation will be computed.

So, as a result as we have seen in case of batch normalization, that the dimensionality of a mean vector or the dimensionality of the standard deviation vector is same as the number of channels because for every channel you compute one mean and one standard deviation where as in case of layer normalization because the mean and standard deviation are computed from all the features all the feature maps or all the channels coming out of a single example.

So, here the dimensionality of the mu vector or the dimensionality of sigma vector the standard deviation vector that will be same as the number of channels that you have right. So, using the same example or example case that we were discussing previously.

(Refer Slide Time: 08:46).



So, here what we had is N is the number of training samples per batch and C is the number of channels. So, here what we are doing is for every training example i am computing mean and standard deviation and this computation is done by considering the features of all the channels we are computed from the same example.

So, mathematically it will be put like this that mu N will be computed as these are the features from example N and you have channel i i th channel and jk represents this is the index of the feature in that particular N th example i th channel.

So, that the way you compute mu is just x N i j k take the summation over i j and k and then you divide that with c into w into h that is the number of total number of data you have within that particular layer. So, this is how you compute mu N. Similarly you compute sigma N square which is the variance over the same data over the same layer and your normalization will be x hat will be x minus mu N upon square root of Epsilon plus sigma n square.

So, as we told before that this Epsilon is a very small positive constant and it is introduced to ensure that I do not come across any situation like division by zero which makes your division unstable. So, this is how your layer normalization works that in case of layer normalization the way you compute mean and standard deviation is that you get all the channels from a particular training example and over those channels you compute

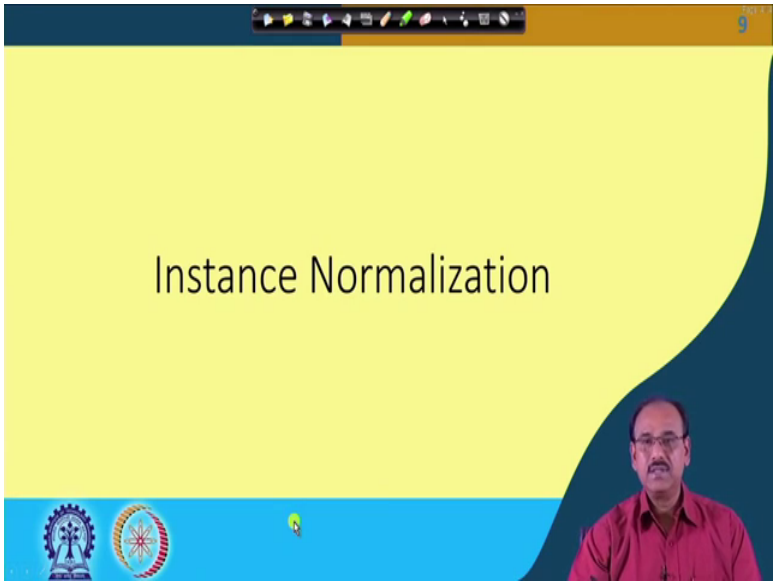
mean and standard deviation and using those mean and standard deviation you normalize your data that is x hat.

So, again over here as in case of batch normalization i have to go for reparameterization; that means, i have to compute say yi which is same as some gamma times x i hat plus beta. Where this gamma and beta are now mu parameters telling you what is the distribution of the data. And this gamma and beta are tunable and they can be tuned in the same gradient descent procedure along with the weight vectors of the neural network as we have discussed in case of batch normalization technique.

So, you find that in this layer normalization in case of batch normalization i had to wait for all the batches i mean all the samples training samples belonging to the batches to be processed before i can compute the mean and standard deviation of a particular channel. So, here i will compute mean and standard deviation of a particular channel when it is batch normalization, but in case of layer normalization i have to compute the mean and standard deviation of the channels for a particular training example.

So, here as i am not waiting I do not have to wait for processing of all the samples all the training samples in the batch. So, this can be utilized in other kind of operations where your batch processing is difficult, like in case of recurrent neural network or even simple stochastic gradient descent where the network is stained with every training example right not over batches. So, this is what is layer normalization.

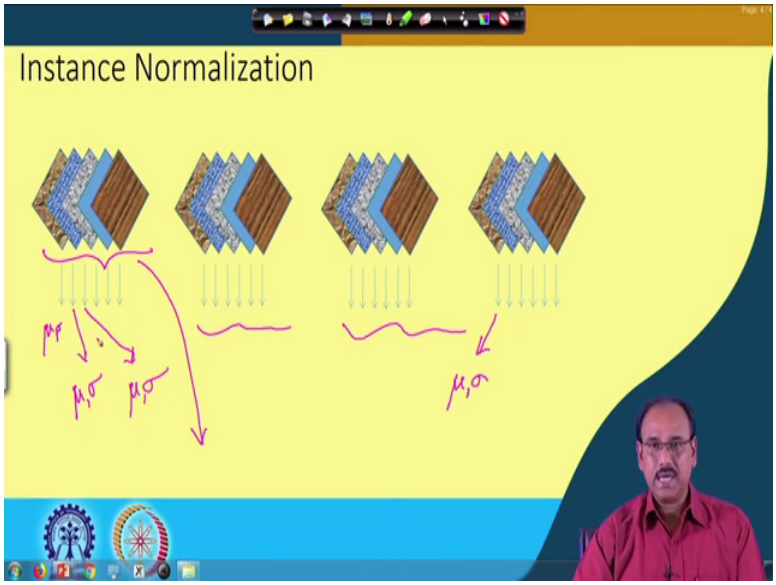
(Refer Slide Time: 12:57).



The other kind of normalization techniques that we have is what is instance normalization.

So, even in case of layer normalization you find that you are computing mean and standard deviation by considering all the channels that you get out of a training example. In case of instance normalization I will not even consider all the channels of a training example, but for every individual channel will be normalized with respect to that channels mean and standard deviation. So, the operation is something like this.

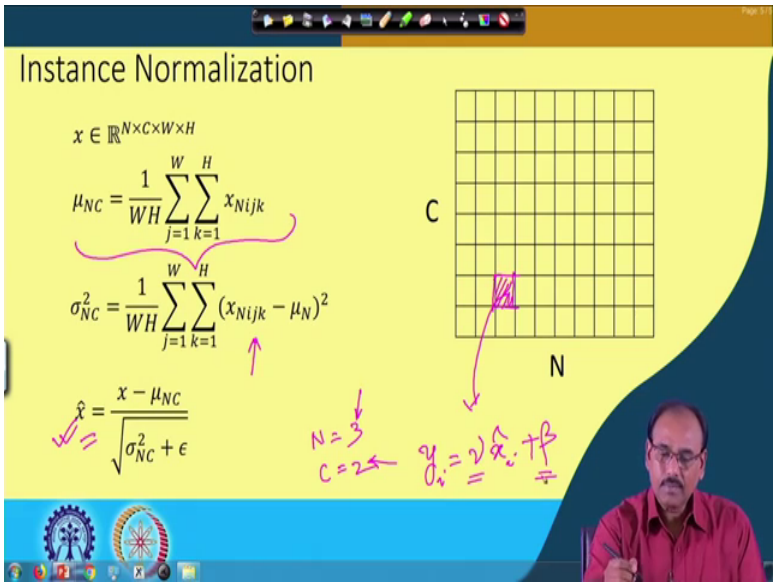
(Refer Slide Time: 13:28).



So, here you find that for every channel over here you compute one mean and standard deviation. For this channel for this channel you compute another mean standard deviation, for this channel you compute another mean and standard deviation, similarly over all other cases. So, for every channel you compute one mean and standard deviation. So, in case of layer normalization all of them taken together gives you one mean and one standard deviation.

Whereas, in case of instance normalization for every individual channel you are computing mean and standard deviation and with respect to that mean and standard deviation you are normalizing the features in that particular channel.

(Refer Slide Time: 14:22).



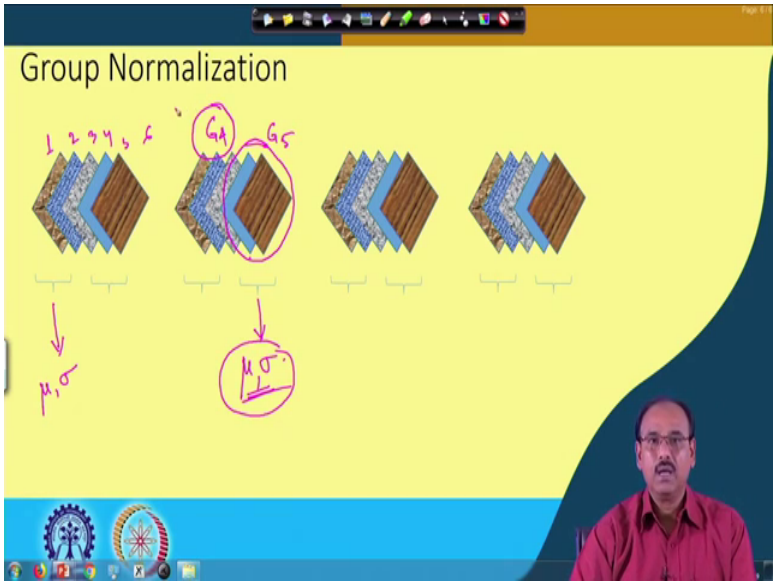
So, mathematically as we have seen before this operation can be put like this that again i have N number of samples in a batch i have C number of channels and every channel is of width W by H.

So, you find that now mean and standard deviation of every channel is computed over the channel itself right. So, here if I consider say for example, N is equal to 3 and C equal to 2; that means, for third example and second channel i take third example over here and second channel. So, I compute the mean and standard deviation for this particular channel only. And this is the expression for that right. Similarly i compute the variance for width the features within this channel only I do not consider the features belonging to other channels for computing this mean and standard deviation and with respect to that you normalize the features in that particular channel only.

So, for standard for normalization of the features belonging to different channels are to compute mean and standard deviation pertaining to that particular channel only I do not consider any other channel to compute that mean and standard deviation. And with respect to that you are normalizing the features and again here you find that the same reparameterization is applicable that is here again i have to compute Y i which is equal to gamma times x i hat plus beta where this gamma and beta will be tuned using the back propagation learning algorithm along with other parameters of the network.

So, this is what is instance normalization and we have also talked about other normalization techniques that is known as group normalization.

(Refer Slide Time: 16:19).

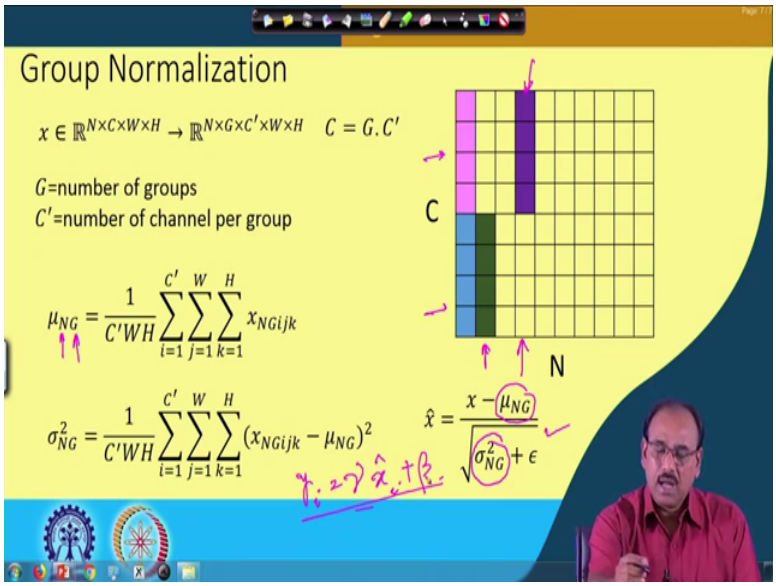


So, what you do in case of group normalization? In case of group normalization the different channels are grouped into different groups say as over here. May be channel number 1 channel number 1 2 they are put in one group, then 3 and 4 they will be put another group, 5 6 they will be put in another group and so on.

And for each such group of an example you compute the corresponding mean and corresponding standard deviation. So, similarly for this group you compute a mean and you compute a standard deviation. And when you normalize the data you normalize the features belonging to that group only with respect to the corresponding mean and standard deviation. So, when I compute this mean and standard deviation for this group let me call it is a group G 5 and this is say group G 4 and mean and standard deviation is computed from the features or from the channels in G 5 only.

So, when I go for normalization with these feature with these mean and standard deviation we will normalize the features of the channels belonging to group 5. We will not use this mean and standard deviation to normalize the features or the channels belonging to group 4.

(Refer Slide Time: 17:49).



So, again mathematically this group normalization can be put like this. So, here you find that if every channel or the number of groups that you have is G and say C prime is the number of channel per group; that means, your total number of channels now becomes C equal to G into C prime. Where G is the total number of groups and C prime is the number of channels per groups.

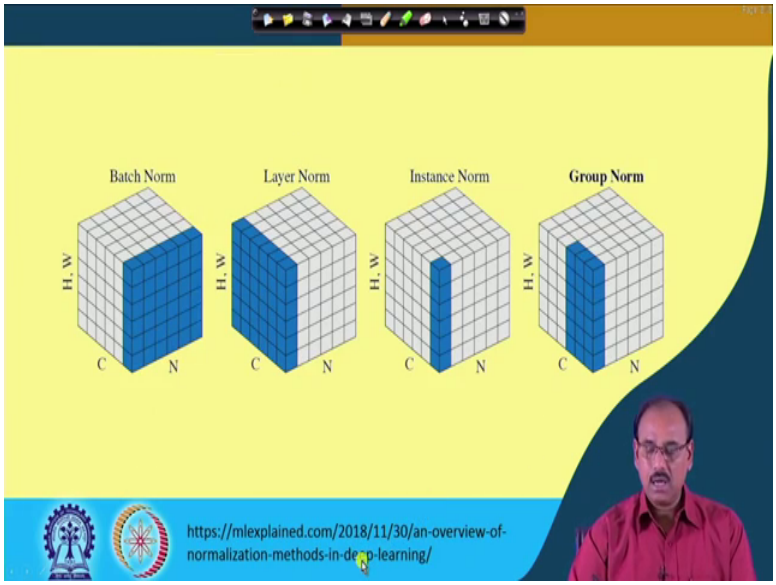
So, that is that is what has been shown over here pictorially. So, this is what. So, this is 1 group this is another group this is another group this is another group and so on. So, you find that this group every group consists of a number of particulate a number of channels corresponding to a particular example. And here again when you compute the mean and standard deviation you compute the mean and standard deviation for that group only ok.

So, here you have mean and standard deviation for a particular training example and then a particular group as has been shown over here pictorially. And then you have normalized the features within the corresponding group following the same procedure with respect to the mean that you have computed over the group and with respect to the standard deviation that you are computed over the group.

Of course, here again the same process of reparameterization that is projecting back the normalized data as y i is equal to gamma times x i hat plus beta. That is also applicable in this case. And this gamma and beta will be updated or tuned using the back propagation learning algorithm along with the parameters or the weight vectors of the network. So,

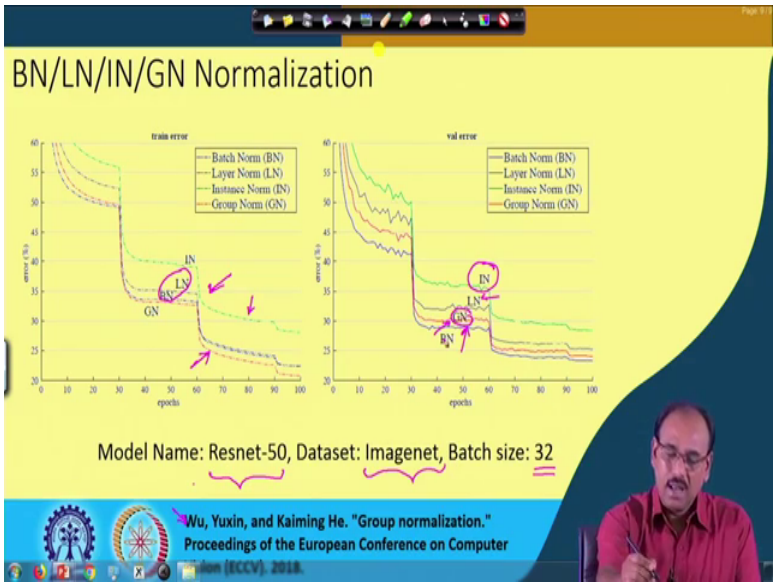
these are the different normalization techniques that can be employed to take care of the covariate shift problem.

(Refer Slide Time: 19:53).



Now, let us see that how these this is another way of representing what is Batch Norm what is Layer Norm what is Instance Norm and what is Group Norm i mean the same figure, but put in a 3 dimensional form where actually W and H has been put as 1 dimension. In earlier discussion what we have done is we are put it in 2 dimension assuming that every cell within that 2 dimensional array is a matrix of W by H. So, it depicts the same thing

(Refer Slide Time: 20:30).



So, now, to look at the performance of these different normalization techniques you find that there is an output which is taken from this particular paper has given over here by Wu Yuxin and Kaiming He.

So, they have given the relative performance of this different normalization techniques. Both on the training data as well as on the validation data. So, what has been plotted is the training error as well as the validation error. So, on the left hand side and this was experiment was done on Resnet 5 and the data set is same as Imagenet and the Batch size that was taken was 32 that is 32 example staining examples in a batch ok.

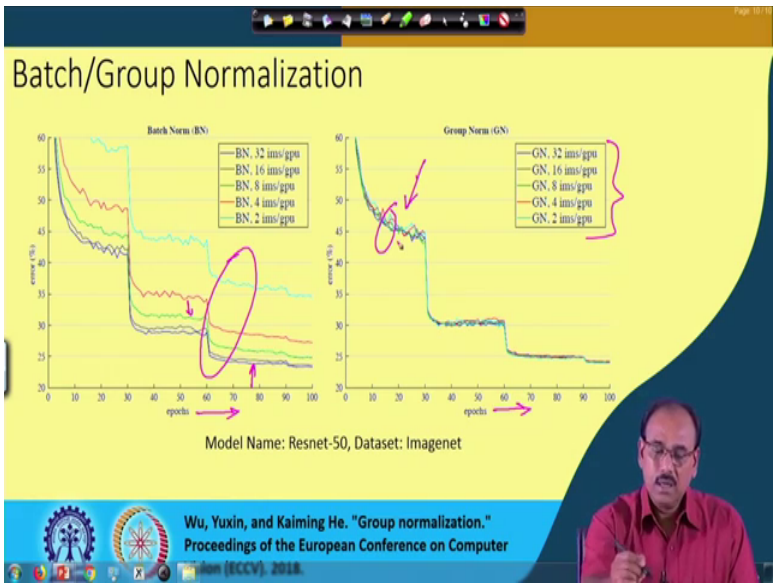
So, the training error for different normalization techniques has been shown over here. So, here you find that during training it is the group normalization techniques which performs best among all of them in this particular experiment. And the instance normalization which is over here that performs the worst and the batch normalization and layer normalization the performance is somewhere in between.

When it goes to validation error, so validation process is when you train a neural network or train a deep neural network your first step is you have to train the neural network using the training data. So, all the data set you have to put into three different partitions one partition is the training data set another partition is validation data set and another partition is test data set. So, you train that all these data sets are mutually exclusive.

So, you have to train your network using the training data set and then you see the performance of the trained network using the validation data set. So, there you get what is known as validation error and it is expected that a network which is properly trained and performance validated should perform should give a satisfactory performance on the test data set as well. So, here what is shown is that training performance with respect to training error and the performance with respect to validation error.

So, when it comes to validation error here you find that the batch normalization. So, far as this particular experimental result is concerned the batch normalization has performed the best. Group normalization is slightly worse than batch normalization instance normalization as in case of training error it is performing worst layer normalization is somewhere in between. Though you find that here your group normalization performs what's that then batch normalization, but the group normalization gives you other advantage as has been experimented.

(Refer Slide Time: 23:56).

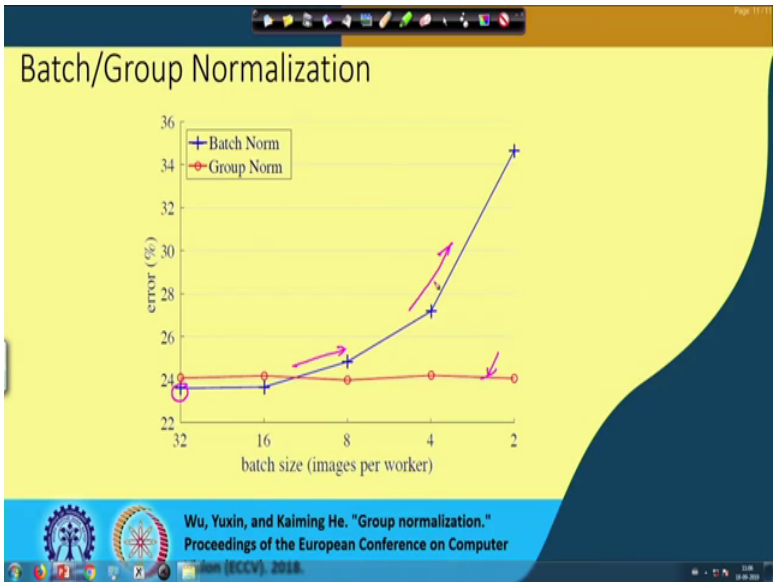


So, here you find that the Batch Normalization versus Group Normalization for different batch size ok. So, when you perform only batch normalization you find that as you vary the batch size the error percentage varies right. So, when your batch size is only 2 i mean the batch size has varied from 32 to 2 and you find that the error or the batch normalization performs best as you have seen before when your batch size is 32 right. When the batch size is two the performance is worse that is the error rate has increased.

So, here you find that what this diagram depicts is and these are the different errors that you get within different epochs that is the stages of training. So, what this diagram depicts is that the performance of the network with batch normalization depends upon the batch size or the error rate depends upon the batch size. On the right hand side what is shown is if I use group normalization then of course, as desired we as the training proceeds with the number of epochs your error rate reduces, but the error rate is more or less independent of the batch size that you are using.

So, if your batch size is the 2 or batch size is 4 or 8 or even 32 the error rate is more or less than. So, this shows that your group normalization is independent of the batch size the performance of the group normalization is independent of the batch size whereas, performance of the batch normalization depends upon the size of the batch.

(Refer Slide Time: 26:09).



That is also shown with the help of another plot which has shown over here you find that here you vary the batch size and for different batch size of compute the error.

So, as this diagram again shows that when you go for group normalization which is this red plot with different batch size the error rate remains more or less same, but when you go for batch normalization when your batch size is quite large say 32 or so, performance of the batch normalization is better than group normalization, but as the batch size reduces the performance of batch normalization becomes worse than group normalization and the performance becomes even worse as you reduce the batch size.

So, these are the performance of different normalization techniques that we have discussed and you also find that this when you go for normalization the normalization technique not only takes care of the covariate shift of the training data. It also imposes some sort of regularization the regularization which prevents the network from over fitting.

We will discuss about that in details later, but just how this normalization techniques prevents the over fitting or it imposes some sort of regularization on the network is that we have seen when we have talked about de noising auto encoder that in order to regularize the network or prevent the network from over fitting. What you do is you input a noisy data during the training operation.

Here when you go for batch normalization you are computing the mean and standard deviation over a batch of data which is not the actual mean and standard deviation over the inter set of data. Or in other words you can say that the mean and standard deviations that you are computing those are actually noisy mean and standard deviation and because of this noise you are training your neural network with the noisy data considering that the mean and standard deviation that you have computed is noisy.

So, this (Refer Time: 28:32) says some short of regularization as a result the network does not over fit on to the training data. So, you find that this is a short of side effect I mean normalization is not really meant for regularization, but it imposes it gives some sort of regularization. So, with this we will stop here today we will continue with other topics from our next class.

Thank you.