Course Name: Machine Learning and Deep learning - Fundamentals and Applications

Professor Name: Prof. M. K. Bhuyan

Department Name: Electronics and Electrical Engineering

Institute Name: Indian Institute of Technology, Guwahati

Week-11

Lecture-40

Welcome to NPTEL online course on machine learning and deep learning fundamentals and applications. In my last class, I explained the concept of two very popular CNN architectures. One is the Lenet 5 and another one is the AlexNet. Lenet 5 was developed in 1998. That is for the automation of the banking industry. And after this, I considered one data set that is the ImageNet data set.

So in the ImageNet data set, there are 1000 classes and the problem is the image classification problem. So for this, there is a computer vision competition that is the ILSVRC competition that already I have discussed about this competition. And winner of this competition in 2012 is the AlexNet. So I discussed the concept of the AlexNet.

So in the AlexNet, we are considering multiple convolutional layer and also the max pooling layer. So that concept also I have discussed in my last class. So today I am continuing the same discussion. And for this discussion, today I am considering another three popular CNN architectures. One is the VGGNet.

After this, I will discuss the concept of the GoogleNet. And finally, one very important network that is the residual network. So let us discuss about these three important CNN architectures. So the CNN architectures, one is the VGG16, GoogleNet and the residual net. So in my last class, I discussed the concept of the LeNet 5.

So this was developed in 1998. That is that that was developed for the automation of banking industry. And in this case, the problem is the character recognition. So input image is the grayscale image of the size 32 x 32. And after this, the convolution operation is done.

So C1 is the convolution operation and six feature maps were considered. After this, the pooling operation was performed. And again the convolution operation. After this, again the pooling operation is done. And finally, the fully connected layers were considered.

And you can see the final output layer. Finally the Gaussian connections were considered for the final classification. So this concept already I have explained. So one important point is the error rate as low as 0.95 % on the test data that is achieved by the LineNet 5.

So that is one important point. After this, I considered one image data set that is the ImageNet data set and that is considered for ImageNet large scale visual recognition salience. And these are the samples of the ImageNet data set. So altogether 1000 image classes are considered. So 1000 different classes are available in this data set.

That is the ImageNet data set. After this, I discussed the concept of the LXNet. So in the LXNet, the concept is the repeated convolution and the max pooling operation. So here you can see the input is 227 x 227 x images. That is the RGB image.

And after this repeated convolution, overlapping, max pooling, convolution, overlapping, max pooling operations are done. And finally, these fully connected layers, you can see these 4096 nodes in the fully connected layers. And finally for classification, the layer is the output layer. That is the 1000 nodes, softmax layer. So output layer is the softmax layer having 1000 nodes, because the problem is the classification problem and 1000 classes are available in the ImageNet dataset.

So this concept already I have explained. So if you see this, this ILSVRC winners, so we discussed about the winner AlexNet that is that was developed in 2012. After this, the next one is very popular is the VGG network. In the VGG network, 19 layers are available and the error rate is 7.3. So you can compare the performance of the VGG network with the AlexNet. In the AlexNet, we considered 8 layers, but in the VGG, 19 layers are available. So error rate is 7.3 as compared to 16.4. After this, the next important network is the Google network. That is the GoogleNet. That was also the winner in the competition ILSVRC 2014 competition. And it is a very deep architecture because 22 layers were considered and with the help of these 22 layers, 6.7 error rate that was achieved with the help of the GoogleNet.

So we will discuss about the VGGNet and the GoogleNet. And finally, the residual net that was the winner of the 2015 ILSVRC competition. And you can see the number of layers significantly increased to 152 layers. So it is a very deep architecture, the residual net and the error rate as low as 3.57 is achieved by the residual network.

So that means we are moving towards the deep networks that is the deep architecture. So AlexNet only 8 layers we considered, VGGNet 19 layers, the GoogleNet 22 layers and finally residual network. This is a very deep architecture, 152 layers are available.

So now I will discuss the concept of the VGGNet. So VGG16 that is the ILSVRC 2014 first runner up.

So what is VGG? That is actually the Visual Geometry Group, Oxford University. So that is actually the VGG, the Visual Geometry Group, Oxford University. So now let us discuss about the VGG16. So this was developed in 2015. And this is the runner up in the competition ILSVRC 2014.

So it is significantly deeper than the AlexNet and altogether 140 million parameters. So that is the summary of the VGGNet. So let us see the structure of the VGGNet. So this is the structure of the VGG16.

The input is a RGB image. The size of the image is 224 x 224. So we are considering 3 channels RGB channels. And here you can see this black colored box that represents the convolution. The red color box that corresponds to the max pooling operation. This blue color box that corresponds to the fully connected layers.

Similarly for classification of 1000 classes, the softmax is considered. So here you can see we are considering the ReLU activation function. And it is nothing but the repeated convolution operation and the pooling operation. And finally we are considering the fully connected layers. And for the final classification, the output layer is the softmax activation function that is considered for 1000 classes.

So you can see that we are doing the convolution. So it is the size is 224 x 224 and 64 kernels are considered. So like this we are considering repeated convolution operation and the max pooling operation. So how actually it is different from the AlexNet. So that concept I am going to explain.

So what is the difference between the VGG16 and the AlexNet. So if you see the structure of the VGGNet. So the input image is available. After this we are considering 3 x 3 convolution.

64 kernels are available. After this again 3 x 3 convolution, 64 kernels are available. After this the max pooling operation which stride 2. So stride 2 is considered. After this again 3 x 3 convolution and number of kernels are 128. Again 3 x 3 convolution, number of kernels are 128.

Again pooling operation that is with the stride 2. So like this we are doing the convolution and the pooling. And for the fully connected layers 4096 nodes are available. So 3 fully connected layers. So one is the 4096 and final fully connected layer has 1000 nodes because we are considering 1000 classes of the ImageNet dataset.

So this activation function the Softmax activation function is considered. So this is the structure of the VGGNet. Now here the one important point is we are considering only 3 x 3 convolution filter. In case of the AlexNet we are considering variable size convolution filters. But in the VGG we are considering 3 x 3 convolution filter stride 1 and also we are doing the pad 1.

The padding is 1 that is the 0 padding because we have to consider the same size of the feature map as that of the input size. So for this we are considering the 0 padding so that I will be getting the feature map which is of the same size as that of the input image. So that is why the 0 padding is considered. And you can see the 2 x 2 max pooling we are considering and the stride 2 is considered. So the main difference between this AlexNet and the VGGNet.

In the AlexNet we considered 8 layers but in the VGGNet 16 to 19 layers were considered and also the VGGNet gives 7.3 the error in the competition that is the ILSVRC 2014 competition. So why we are considering the 3 x 3 convolution instead of considering variable size convolution filters. So that concept I am going to explain in my next slide. So we are considering smaller filters that is the 3 x 3 convolution we are considering and we are considering the stack of 3 3 x 3 convolution stride 1.

So instead of considering 7 x 7 convolution filters in VGGNet 3 3 x 3 convolution filters are considered in place of 1 7 x 7 convolution filter. Because 3 3 x 3 convolution filter has the same effective receptive field as 1 7 x 7 convolution layer. So that means I am repeating this in VGGNet 3 3 x 3 convolution layers are considered instead of 1 7 x 7 convolution layer. Because the 3 x 3 convolution layer has the same effective receptive field as that of 7 x 7 convolution layer. So what is the effective receptive field of 3 x 3 convolution stride 1 layers as compared to 7 x 7 convolution layer.

So one point is the few number of parameters. So if I consider number of parameters so we are considering 3 3 x 3 convolution layers so that is why it is 3 after this the size of the filter is 3 x 3. So that is why it is 3^2c^2 because we are considering c channels per layer versus if I consider only 1 7 x 7 convolution layer. So in this case you can see 7^2c^2. So if you compare these two the first one is this and the second one is this. So you can see the 3 x 3 convolution has less number of parameters as compared to 7 x 7 convolution layers.

So here you can see I am considering 3 3 x 3 convolution filters. So the first one is the 3 x 3 filter the second one is also 3 x 3 and third one is also 3 x 3 the filters. Corresponding to this we are getting this output so this is the output. So this output I can obtain directly by considering the 7 x 7 kernel but we are considering 3 3 x 3 filters the convolution filters but the receptive field will be same. If I consider the 7 x 7 filter we are getting the same receptive field as obtained in 3 3 x 3 convolution layers.

So that is why the first method is more efficient because we are considering 3 3 x 3 convolutional filters instead of only 1 7 x 7 convolution filter. So the number of parameters will be less if I consider 3 x 3 convolution filters. That means we are considering a simple structure by considering the 3 x 3 convolutional layers. So this is the concept of the receptive field. So in summary the input to the network are color images the size is 224 x 224.

The training procedure is very similar to the AlexNet we can consider the backpropagation training the images pass through a stack of convolutional layer. So this is very similar to the AlexNet and actually there are 2 versions of the VGGNet one is the VGG16 and another one is the VGG19. So VGG19 only slightly better but it requires more memory and also we are considering row and column padding that is the zero padding to maintain spatial resolution after the convolution. So the zero padding is done to maintain spatial resolution after the convolution. So there are 13 convolutional layers 5 max pooling layers and the max pooling window size is 2 x 2 and the stride is 2 and this was trained on 4 NVIDIA Titan Black GPUs for 2 to 3 weeks.

So you can see how much time is required for the training of this network that is the VGG network because there are so many parameters for the training. So this is about the VGGNet and the not every convolution layer is followed by max pooling layers in case of the AlexNet we first do the convolution and after this we considered max pooling but in this case not every convolution layer is followed by max pool layer and in the final layers in VGGNet 3 fully connected layers are available. First 2 fully connected layers have 4096 channels and the last fully connected layer has 1000 channels because the problem is the 1000 image classification problem. So that is why the last fully connected layer has 1000 channels. The last layer is a softmax layer with 1000 channels one for each category of images in the image net database and in this case the railway activation function is considered in VGGNet.

So this is the summary of the VGG network and you can see the fundamental difference between the VGGNet and the AlexNet. So mainly we are considering the simple convolution that is the 3 x 3 convolution we are considering instead of variable size convolution layers. So the striking difference from the AlexNet is all convolutional kernels are of size 3 x 3 which stride 1 so that is already I have explained all max pool kernels are of size 2 x 2 which stride 2 that is the case. All variable size kernels as in AlexNet can be realized by considering multiple 3 x 3 kernels this concept already I have explained. This relation is in terms of the size of the receptive field covered by the kernels.

So we are considering the 3 x 3 kernels instead of variable size kernels as used in the AlexNet. So that is the striking difference between the AlexNet and the VGGNet. So the main objective of the VGGNet is keep it deep but keep it simple. So keep it deep because we are considering the deep architecture but we are considering the simple implementation because we are considering 3 x 3 kernels. After this the next important architecture is the GoogleNet ILSVRC 2014 winner.

So now let us see the architecture of the GoogleNet. So it was developed in 2015. So ILSVRC 2014 competition winner and it is significantly deeper than the AlexNet and it has less number of parameters than the AlexNet and main focus is the computational efficiency. This is the main consideration in the GoogleNet. So this is the architecture of the GoogleNet. So here you can see the blue colored box that is for the convolution layer.

The red color box that is for the max pooling layer. This green colored box that is for the fissure concatenation and the yellow color box that is for the softmax layer. So these are considered. So in this network at the 22 layers with the parameters. So we are considering 22 layers with parameters and 27 layers with max pool layers.

Obviously the max pool layers has no tunable parameters. So 22 layers with the parameters and 27 layers with max pool layers. So this is the structure you can see here. So we are considering one particular module. So if you see this module this module this module is called the inception module.

If you see the complete architecture of the GoogleNet. So this module is repeated again and again. So that means this is the fundamental or the basic unit of the GoogleNet. So how many inception module in the GoogleNet. If you see we are counting this is 1, this is 2, 3, 4, 5, 6, 7, 8, 9.

So that means in this network 9 inception module. So these are the basic units. So the entire network is constructed by considering these modules that is the fundamental unit and this is called the inception module. So by considering 9 inception module this Google network is developed. So the fundamental concept of the GoogleNet is that is the architecture.

So 22 layers 9 inception module that is already I have shown. So in this figure also you can see 9 inception module. So this is number 1, number 2, number 3, number 4, number 5, 6, 7, 8 and 9. So 9 inception modules are available. In case of the LXNet or in case of the VGGNet we are considering repeated convolution and the max pooling operation.

But in this case we are not considering that concept. We are considering the efficient inception module. In this case we are considering the efficient inception module. And one important point is no fully connected layers and only 5 million parameters. So this GoogleNet it was the winner in ILSVRC 2014 classification problem.

That is the ImageNet dataset problem and the competition was ILSVRC 2014. So 6.7% was the error rate that time. So now I will discuss about this inception module. So what is the inception module? So the structure of the inception module is like this.

So this is the structure of the inception module. So computing 1 x 1, 3 x 3 and 5 x 5 convolutions within the same module of the network. So if you see the inception module what we are considering the 1 x 1 convolution, 3 x 3 convolution and 5 x 5 convolution it is considered together. And also the max pooling is also considered. I will explain what is the max pooling. But in this case the computation is done for 1 x 1 convolution, 3 x 3 convolution, 5 x 5 convolution within the same module of the network.

So 5 x 5 convolution that is used for covering a bigger area of the image or maybe the feature map. So that is the covering a bigger area. And also if I want to consider the fine details of the image or the feature map. So that is why the small convolution, the small size convolution is considered. So that is why the covers a bigger area at the same time preserves fine resolution for small information on the images.

So that is why we are considering 1 x 1 or maybe the 3 x 3 convolution. So one objective is to cover the bigger area and another objective is to consider the fine resolutions for small information on the images. And that is why we consider different convolution kernels. So one is the 1 x 1 kernel, 3 x 3 kernel and 5 x 5 kernels are considered. So before this convolution this 1 x 1 convolution and again you can see this 1 x 1 convolution was considered to reduce the amount of computation. So I will explain what is this 1 x 1 convolution why actually it is considered to reduce the number of computation that I will be explaining in my next slide.

So this is the inception module. If you see the figure here, this is the inception module that is nothing but the network within a network. That concept is nothing but the network within a network. So we are considering this inception module. We are considering variable size convolutional filters that is 1 x 1, 3 x 3, 5 x 5 because the objective is to consider the bigger area and also to consider the fine details of the image or the feature maps. And this 1 x 1, so this 1 x 1 and this 1 x 1, this is considered for reducing the amount of computations.

So that concept I will explain after some time. And we are also considering the 3 x 3 max pooling operation. So this is the naive inception module. That was the naive inception module. So in this case, we are considering the multiple receptive field sizes. That means we are considering different size kernels, 1 x 1 convolution, 3 x 3 convolution, 5 x 5 convolution.

But we are not considering the 1 x 1 convolution that I have shown in my previous slide. So this 1 x 1 convolution, 1 x 1 convolution I have not shown in this figure. So this is the naive inception module. And after this what we are considering the concatenating all the filters output together depth wise. So we are doing the filter concatenation after the convolution. So 1 by if you see the previous layer it is considered after this performing 1 x 1 convolution, 3 x 3 convolution, 5 x 5 convolution and also we are considering the 3 x 3 max pooling.

And finally, the concatenating all filter outputs together and depth wise that is the filter concatenation. So this is the naive inception module. So what is the problem with this inception module? So you can see here. So what is the problem of this inception module? The problem is the high computational complexity. So that concept I am going to explain in my next slide what is the high computational complexity.

Suppose the previous layer has the size 28 x 28 and 256 kernels are available. And you can see we are considering different kernels 1 x 1 convolutional kernels, 3 x 3 convolutional kernels, 5 x 5 convolutional kernels. So 1 x 1 convolutional kernels having 128 channels, 3 x 3 convolutional and 192 channels, 5 x 5 convolution, 96 feature maps or the channels we are considering and after this we are considering the filter concatenation. So corresponding to this structure what will be the output volume size for 1 x 1 convolution we are considering 128 channels. So corresponding to this the output volume size will be 28 x 28 x 128. So size will be 28 x 28 and we are considering 128 channels or the feature maps that is corresponding to 1 x 1 convolution.

After this the 3 x 3 convolution. So 129 channels that means 192 feature maps and corresponding to this the size will be 28 x 28 x 129. After this we are considering the 5 x 5 convolution. So 96 channels are available that means the 96 feature maps and output volume size will be 28 x 28 x 96. And if I consider the pooling layer that is the 3 x 3 pooling layer and if I consider 3 x 3 pooling layer the size will be 28 x 28 x 256 because we are considering 256 channels. So what is the output size after the filter concatenation? So the output size will be we will get like this.

So from all this we can compute this and the size will be 28 x 28 x 672. That will be the output size after filter concatenation. That is for the this inception module and how many convolution operations corresponding to this inception module you can calculate this one. So for 1 x 1 convolution because we are considering 128 feature maps. So it is 28 x 28 x 128 x 1 x 1 x 256.

So that is the number of convolution operations corresponding to 1 x 1 convolution. Again 3 x 3 convolution corresponding to this 192 channels we can determine 5 x 5 convolution corresponding to 96 channels we can determine and total operations will be

854 million operations. So huge number of operations corresponding to this inception module. So to reduce this computational complexity we are considering depth 1 x 1 convolution before this layer.

So I will show that one. So it is a very expensive computation. The pooling layer also preserve feature's depth which means the total depth after the concatenation can only grow at every layer. So that means the computational complexity is very high. So to consider this issue we are considering the one bottleneck layer that is actually 1 x 1 convolution we are considering. So that is the objective of the bottleneck layer.

So that is nothing but the 1 x 1 convolution we have to perform to reduce the feature depth. So now I will introduce the 1 x 1 convolutional layer before all these layers. So we are considering this 1 x 1 convolutional layers to reduce feature depth. So you can see we are considering this 1 x 1 convolution this also 1 x 1 convolution we are considering to reduce feature depth that means we are reducing the computational complexity. So corresponding to this inception module what will be the computational complexity.

So how many operations that also we can determine. So corresponding to this inception module the number of convolutional operations we can determine. So corresponding to this 1 x 1 convolution 64 channels are available. So you can determine the number of convolution operations. Again another 1 x 1 convolution for this also you can determine the number of operations.

So like this you can determine all the convolution operations for all the layers. So you can determine all the operations like this. And finally we are having this total the total is 353 million operations. Earlier in the naive version we had 854 million operations. But because of this 1 x 1 convolution that is the bottleneck layer the number of operations is significantly reduced.

So now it is 353 million operations as compared to 854 million operations in the naive version. So this is the structure of the inception module. So that is why this inception module is considered. So 1 x 1 convolution is one important operation in this inception module to reduce the number of operations.

So this is the structure of the Google net. So already I have explained the concept of the inception module. So we have 9 inception module. So already I have shown the 9 inception module.

Also there are two auxiliary classifiers. If you see this is the final classifier. This is the main classifier. So this is the main classifier. And also there are two auxiliary classifiers. So this is the auxiliary classifier.

And this is also another auxiliary classifier. So what is the function of this auxiliary classifier. So there are two auxiliary classifiers and one main classifier for the final classification. If you see this architecture the depth of the architecture is very high because if you see the from the beginning to end in this case there are 22 layers. So that is why the depth of the architecture is very high. So during the back propagation the problem is the vanishing gradient problem.

So the concept of the vanishing gradient problem I have explained. So we have to train this network and because we have so many number of layers the problem will be the vanishing gradient problem. So these auxiliary classifiers are considered to consider this problem the problem of the vanishing gradient problem. So I will explain one by one why actually we are considering the auxiliary classifiers. So due to the large depth of the network ability to propagate gradient back through all the layers was a concern.

That is the problem of the vanishing gradient because the number of layers are more in this GoogleNet. That means the depth of the network is very high. So that is why the problem of the vanishing gradient. Auxiliary classifiers are smaller CNNs put up top of the middle inception modules. So if you see these auxiliary classifiers are put with the inception module at the top of the inception module.

So that concept I have written here. Addition of auxiliary classifiers in the middle exploits the discriminative power of the features produced by the layers in the middle. So that is also one important point because we have to consider the discriminative power of the features produced by the layers in the middle. That also we are considering. So during the training, the laws of the auxiliary classifiers are added to the total loss of the network.

If you see this network, during the back propagation, we have to consider the loss. So now we have to consider the loss corresponding to the main classifier and corresponding to the auxiliary classifiers because we have two auxiliary classifiers. So for the formulation of the loss function, we have to consider the losses of the auxiliary classifier along with the main classifier. So all these losses we have to consider together the loss in the auxiliary classifiers and also the main classifier that is the concept. So during the training, loss of auxiliary classifier are added to the total loss of the network.

The losses from the auxiliary classifiers were weighted by 0.3 and auxiliary classifiers are discarded at inference time. So that means these auxiliary classifiers are considered only during the training, but for the final testing, these auxiliary classifiers are not considered. So with the help of these auxiliary classifiers, we can consider the issue. The issue is the vanishing gradient problem.

So that is the objective of the auxiliary classifier. This is nothing but a simple CNN architecture. So the summary of the GoogleNet is, so in this case the deeper network, it is a deep network with computational efficiency. The main point is the computational efficiency because we are considering the inception module and also we are considering the bottleneck layer to reduce the number of computation. And it has 22 layers, efficient inception module is available. And one important point is there is no fully connected layers and number of parameters are less than the AlexNet and it was the winner of 2014 ILSVRC classification problem.

So this is the summary of the GoogleNet. So main point of the GoogleNet is the computational efficiency. So that is the main or the salient point of the GoogleNet. So how to reduce the computational efficiency by considering the inception module. In the inception module, we have considered the bottleneck layer that is nothing but the 1 x 1 convolutional layers. So because of this number of operations are significantly reduced.

Also the auxiliary classifiers are considered to consider the problem of the vanishing gradient. So this is the concept of the GoogleNet. Finally I will now discuss the architecture of the residual network. So in the residual network, 152 layers are available, very deep network. And it was the winner of the competition ILSVRC 2015.

And you can see the number of layers, it is 152 layers as compared to 22 layers, 19 layers. So it is a very deep architecture. So what is actually this residual network architecture? So let us discuss about this residual network. So it was developed in 2015. Extremely deep network, 152 layers are available. So it is very difficult to train this network because it is a very deep architecture.

So the problem is the vanishing gradient problem or the exploding gradient problem. Because the architecture is a very deep architecture. So these two problems will come, the problem of the vanishing gradient and the exploding gradient. So that is why that is why they considered the concept of the residual learning framework. So what is this residual learning framework that I will be explaining now.

Here you can see the architecture of the residual network, which was the winner of the ILSVRC 2015 classification problem. And here you can see this is the structure. So you can see this is the input. And after this the convolution operation, pooling operation, convolution operation, pooling operation, these operations are considered. And for the final classification, the softmax activation function is considered.

And the fully connected layer has 1000 nodes, because the problem is the classification of 1000 classes of different images. So that is why the fully connected layer has 1000 nodes for 1000 classes. And here you can see this is actually called a skip connection. The skip connection. So what is the need of the skip connections? So I will explain, because this is a very deep network, the problem is the vanishing gradient problem.

So this concept I explained during the discussion of the vanishing gradient and the exploding gradient. So in the figure I have shown again, I have shown the training error with respect to number of iterations, and the testing error with respect to number of iterations. And this is the comparison between two architectures. One is the 20 layered architecture, another one is the 56 layered architectures. It is expected that a 56 layered architecture can give less training error or the testing error as compared to a 20 layered architecture.

But practically it is not true. In the figure also I have shown the 56 layered architecture gives more error as compared to 26 layered architecture during the training and during the testing. In both the cases you can see the comparison between a 20 layered architecture and a 56 layered architecture. So the 56 layered architecture gives more error during the training and during the testing as compared to a 20 layered deep architecture.

This is because of the problem of the vanishing gradient. So that problem already I have explained in my second class. So I am not going to discuss again. So how to consider this problem in case of the residual network. So in the residual network, this problem is considered by considering the skip connection.

So what is the skip connection? Suppose I am considering one network. So this is the input and this is the output. So after some operations like convolution pulling operation, I am getting the output. So these are operations some operations. So I am getting the output and suppose this output is the H(x) that is the output corresponding to the input after these operations.

But I can bypass all these operations and I can connect like this that is the skip connection. So this is the skip connection. So directly suppose input is x. So directly this input I can bypass. Bypass means bypassing all the operations. So these operations I can bypass.

So that means we are considering the residual component the residual is F(x) that is nothing but H(x)-x that is the residual. So here x is the input and this x is bypass and it is directly bypassing this all these operations and it is directly connected to the output by considering the skip connection. So the residual will be F(x) that is nothing but H(x)-x.

So I am explaining it again what is the residual F(x). So let us move to the next slide. So in the figure actually I am showing two figures. One is the plane layers another one is the residual block. So input to the if I see this figure the first figure the input is x after this we are considering some convolution operation ReLU convolution and we are getting the output output is H(x) that is corresponding to the plane layers. But if I consider the residual block so you can see the input is x and we are doing the all the operations convolution ReLU after this suppose we are getting the F(x) and this x is bypass by bypassing this operation the convolution operation ReLU operation convolution operation. So that is nothing but the identity we are getting and this is added with the F(x), F(x) is the residual so F(x) + x and after adding this I am getting H(x).

So after this I am getting the H(x). So H(x) is nothing but F(x) + x. So that is the H(x) that means we are bypassing the convolution and the ReLU operations by considering this skip connection. So this is the skip connection. So if I consider the traditional CNN then the H(x) will be equal to F(x).

So but in this case what we are considering F(x) is added with x to get H(x). So that is why H(x) = F(x) + x. So x is directly taken to the output and F(x) I am computing by considering these layers and finally these are added F(x) + x it is added after this the ReLU is considered and we get the H(x). So there are actually two part one is the direct but another one is the skip connection by skip connection we are considering. So that is the concept of the residual block. So mathematically this concept I can show like this here I am considering two layers. So input this layer is a[l] and we are considering two layers and output of the first layer is a[l+1] and output of the second layer is a[l+2] and we are considering one skip connection.

So this is the skip connection. So if you see this network function so function of the network the input is a[l] so first one is the input is a[l]. After this what we are considering suppose we are applying a linear operation so here we are considering one linear operation. After this we are applying the ReLU operation here so after applying the ReLU operation after applying the ReLU operation we are getting a[l+1] we are getting a[l+1] that is the output of the first layer. After this again we are applying the linear operations so the linear operation is applied after this we are applying the ReLU operation the ReLU operation here and we are getting the output of the second layer the output of the second layer is a[l+2]. So this is the network output corresponding to the direct connection but if I consider the shortcut or the skip connection so this a[l] is directly connected here so by considering the skip connection the a[l] is directly connected to this point.

So in this formulation I am showing one is the direct output and another one is the by considering the skip connection. Now let us see the mathematics so what is z[l+1] so if you see here corresponding to this point z[l+1] is nothing but the width w[l+1] is considered that is multiplied with a[l] and bias b[l+1] is considered that is the z[l+1]. So that is actually the this point the red is this that is the linear after this if you consider another linear so that is z[l+2] corresponding to this point z[l+2]. So w[l+2] x a[l+1] + b[l+2] so that is the B is the bias corresponding to this point and w[l+2] that is the that we are considering the linear response. So w[l+2] x a[l+1] that is for the this one the linear one after this we are considering this blue that is the ReLU activation function we are considering. So what is a[l+1] that is the output of the first layer that is nothing but G is the ReLU activation function z[l+1] because the input to the first layer is z[l+1] so z[l+1] is considered and G is nothing but the ReLU operation we are considering and corresponding to the second ReLU the second ReLU is this what is a[l+2]? a[l+2] is nothing but G the input is z[l+2] so G is the ReLU activation function so z[l+2] that we can determine a[l+2] and a[l+1].

And finally this a[l+2] we can determine like this G, G is the ReLU activation function z[l+2] + a[l] so actually you can see here we are considering the skip input so a[l] is the skip input that is the we are connecting a[l] directly to this point this point we are connecting so that is why a[l] is considered along with z[l+2]. And finally we can put this value I am getting this one so that means we are considering the direct input direct input is the a[l] that is the direct input we are considering and also we are considering z[l+2] we are considering z[l+2] we also we are considering to get the a[l+2] that is the case. So this is the concept of the residual network that means we are considering the skip connections to address the problem of the vanishing gradient. So the summary of the residual network is so stack residual blocks so here we are considering all the residual blocks every residual block has 3 x 3 convolution layers that already I have explained so you can see all the 3 x 3 convolution blocks or the convolution operations it is considered. So additional convolution layer at the beginning so one convolutional layer we are considering in the beginning that is the 7 x 7 convolution layer and in this architecture no fully connected layers at the end.

So there are no fully connected layers but for the final classification one fully connected layer is considered having 1000 nodes because the problem is the classification of 1000 classes of different images of the ImageNet database. So this is the summary of the residual network finally the summary of the residual network that is based on the experimental results able to train very deep networks without degrading because in this case the residual connection is considered that is the skip connection is considered to address the problem of the vanishing gradient or the exploding gradient. Deeper networks now a shift lower training error as expected because of the skip connection the training error reduces that is the objective of the skip connections in case of the residual network that means the residual block is considered in case of the residual network. So in summary in last two classes I discussed the architecture of the LeNet-5, AlexNet, VGG, GoogleNet, and the residual network.

So these are all important architectures. In this class I explained the concept of three very popular CNN architectures the first one is the VGG architecture that is the VGGNet. So one important point is that 3 x 3 convolutional filters are considered instead of variable size kernels. After this I discussed the concept of the GoogleNet architecture.

So two important points one is the inception module and another one is the auxiliary classifier. So auxiliary classifiers are considered to consider the problem of the vanishing gradient and also to reduce the computational complexity the inception module is considered.

After this I discussed the concept of the residual network. So one important point is the skip connection. So with the help of this concept that is the skip connection we can consider the problem of the vanishing gradient or the exploding gradient. That is the summary of these three important popular CNN architectures. So let me stop here today. Thank you.