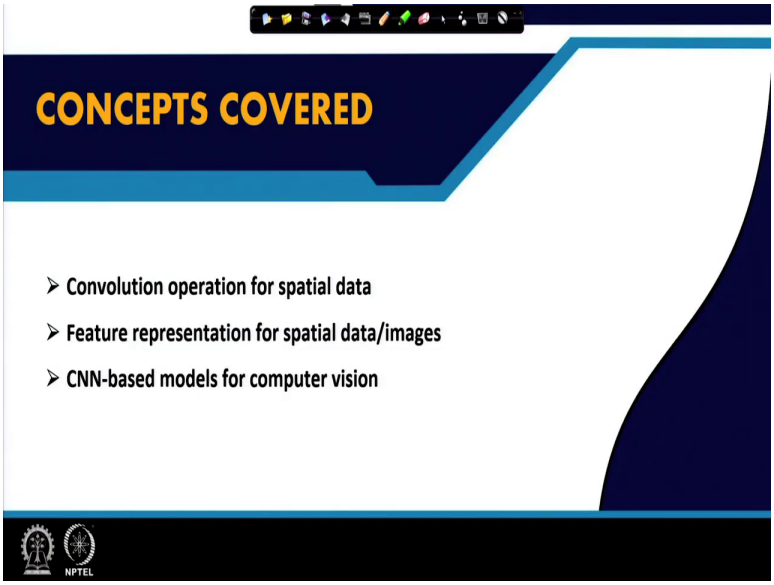
Machine Learning for Earth System Sciences
Prof. Adway Mitra
Department of Computer Science and Engineering
Centre of Excellence in Artificial Intelligence
Indian Institute of Technology, Kharagpur

Module - 02
Machine Learning Review
Lecture - 13
Convolutional Networks of Spatial Problems

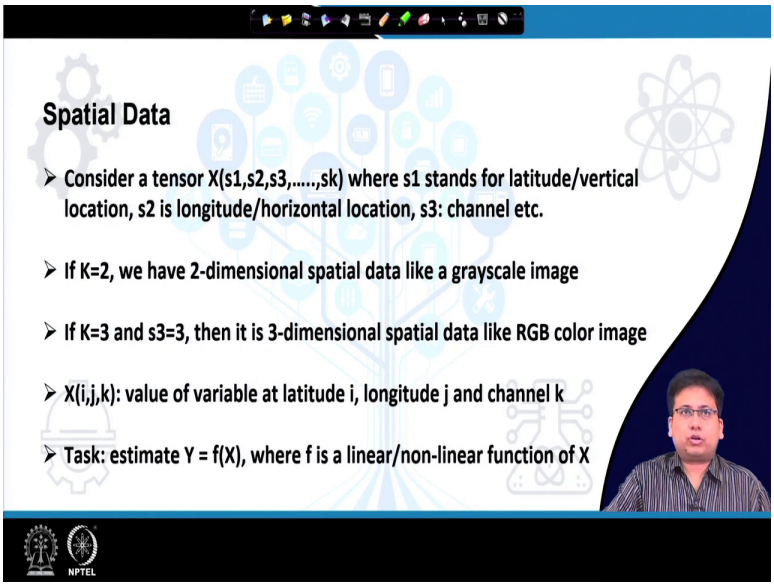
Hello everyone, welcome to Lecture 13 of this course on Machine Learning for Earth System Science. Like you right now we are in Module 2 we are where we are reviewing the various machine learning methods which are relevant for this course. And today we are going to have a lecture on Convolutional Networks for Spatial Problems.

(Refer Slide Time: 00:46)



The concepts that we are going to cover today are as follows; convolution operation for spatial data, feature representations for spatial data or images and finally, the CNN or convolutional network neural network based models for computer vision.

(Refer Slide Time: 01:02)



So, first of all let us talk about spatial data. So, let us consider a tensor like let us say a like a k^{th} order tensor like whose different dimensions of a size $s1, s2, s3, \ldots, sk$. Now, here the first dimension s like which like which is $s1$ this stands for the latitude or the vertical location of a measurement, the $s2$ is like the longitude or the horizontal location, $s3$ is the channel and so on, like the like if there are more dimensions they can also have some like some interpretation.

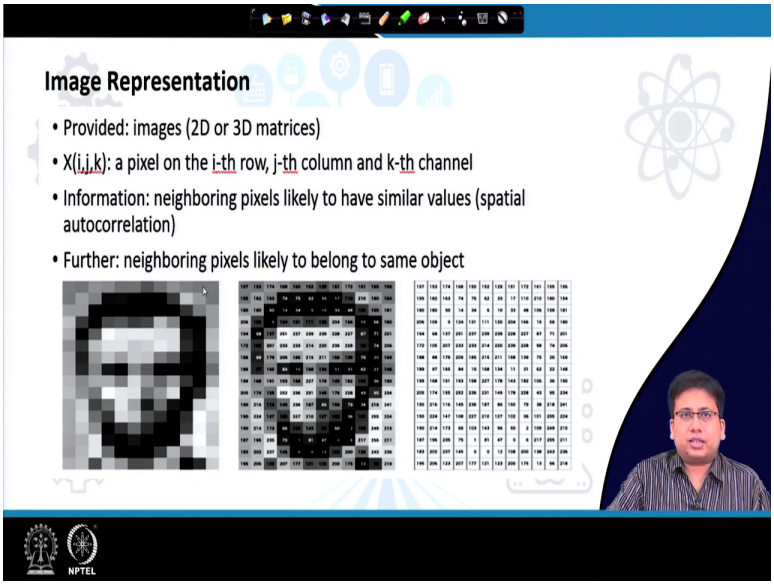
Now, in most cases; however, like this K will be either 2 or 3. So, if $K = 2$ that is then we have only 2 like we have only 2 - dimensional spatial data like a grayscale image. So, we can in that case X is effectively just a matrix, $s1$ is something like its row number and $s2$ is its column number which can also be imagined as latitude or longitude.

Now, if $K = 3$ then it is a 3- dimensional matrix and then like and especially if the number of or if the third dimension like its size is 3 then like we can say it is like a like 3 - dimensional spatial data like RGB or the or like which is a standard color image. So, a color image as you know like it is like 3 grayscale images taken like together, each of those 3 is like they represent the 3 color channels the red channel, green channel and the blue channel.

So, in such a situation in like in case of this 3 - dimensional image if I say $X(i, j, k)$; that means, the value of the variable of interest you can say the pixel value at a latitude i which is basically

the row number, longitude j which is the column number and the channel k. So, now, the task is like you have this image like this 3 - dimensional or 2- dimensional image X which is of course, the collection of all these values which you can which we can call as the pixel values. So, now estimate $Y = f(X)$, where f is a linear or non-linear function of the of X.

(Refer Slide Time: 03:34)



Now, so, suppose like this is a simple example of that. So, suppose we are talking about a pixel. So, then like let I am sorry I mean we if you are talking about a grayscale image so, like this is a typical example of a grayscale image. So, these like these small boxes as you can see these are the different pixels. So, needless to say each pixel is like it has its latitude which is its, we can say its row number starting from the top and its longitude which is the column number may be starting from the left.

And each pixel has a value that value may be any like in case of images which we can see the standard images like which are taken we take with our camera and so on, each pixel takes the value between 0 and 255. So, 0 means like very dark and 255 means very bright or white color. So, like if it is like the those pixel intensities they are stored in this kind of a 2-D matrix every cell of the matrix or every element of the matrix like contains a pixel value. So, if we like actually visualize it by attaching colors to each of these locations based on the corresponding intensity this is what it may look like.

So, as you can see this is probably like a very coarse grained representation of a human being's face, but as we increase the resolution by resolution I mean the number of these rows and columns. Then we see the image getting sharper and sharper and we will like all the different features of the human face, the eyes, the nose, the mouth, the hair, etcetera they will become more and more prominent as we make the image like sharper that is to say of higher resolution.

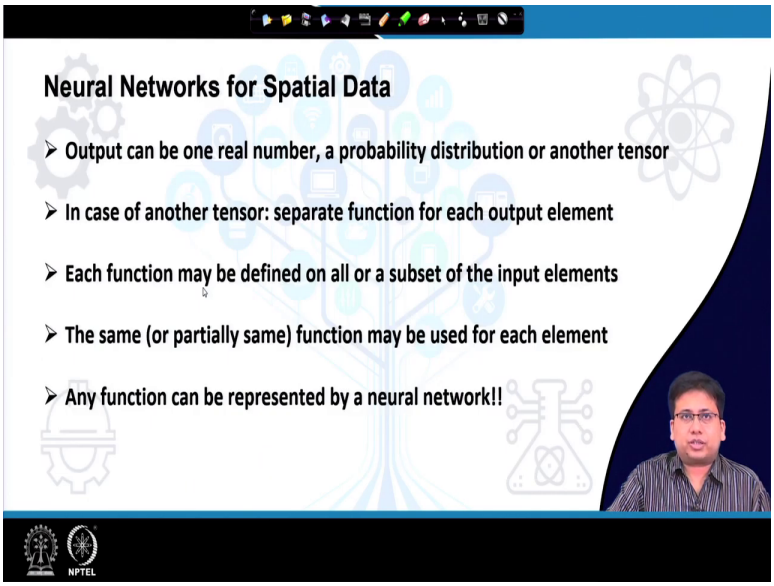
On the contrary, if we keep on lowering the resolution by actually merging these pixels further that is if instead of these 4 pixels if they are replaced by 1 pixel, these 4 pixels are replaced by 1 pixel and so on, then we will see the image getting even more blurred than this and so like in, so, like if the process of making it sharper or more high resolution that is sometimes it is called a up sampling and the process of making it blurred by coarsening the data that is sometimes called as down sampling.

Now, the like we have an additional information in case of most kinds of spatial data including image like visual images like this, the that information is that neighboring pixels are likely to have similar values. That is the spatial autocorrelation property which we have studied in the sometime in module 1. And also further we also know that neighboring pixels are likely to belong to the same object.

So, like in this case for the different objects on a human face are of course, the eyes, the nose, etcetera. Like if it is instead of a human face if it were a general, let us say a scenery then like that different objects could have been say trees, hills, roads, buildings etcetera. So, we can say that most neighboring pixels are likely to belong to the same object that is to say because the objects are coherent structure.

If this is an image and an object is present at a particular location then it is unlikely that the object will be partially present in this pixel, partially present in that pixel, partially present in this pixel and so on it is not like that, usually, the object is a covered by a contiguous block of pixels. So, neighboring pixels are likely to belong to the same object.

(Refer Slide Time: 07:30)



Now, of course, here we are not talking about a visual images per se, but many of when we are dealing with other kinds of spatial data in the which emanate in the domain of earth system sciences there also many of these properties are found to be holding good. In fact, we have especially this property of spatial autocorrelation that we have discussed in our earlier lectures that is very much true in case of earth science data also.

Now, the so, like we already mentioned this $Y = f(X)$, now what these what kind of thing is the your mathematical structure is that Y? So, this Y this can be a one real number, it can be something like a probability distribution over a like over a range or it can be another tensor that is like by tensor like I also mean it can be like a vector or a matrix and so on.

Now, in case it is some kind of a tensor then we can say like that is. So, it will; obviously, be like a collection of many values, say if the input is an of a if input is an image X and the output is another image Y. So, that output image will also have several pixels and the function f that we mentioned like it is like mapping X to one particular pixel. So, similarly like we will have to map the input image X to all the pixels of the output image.

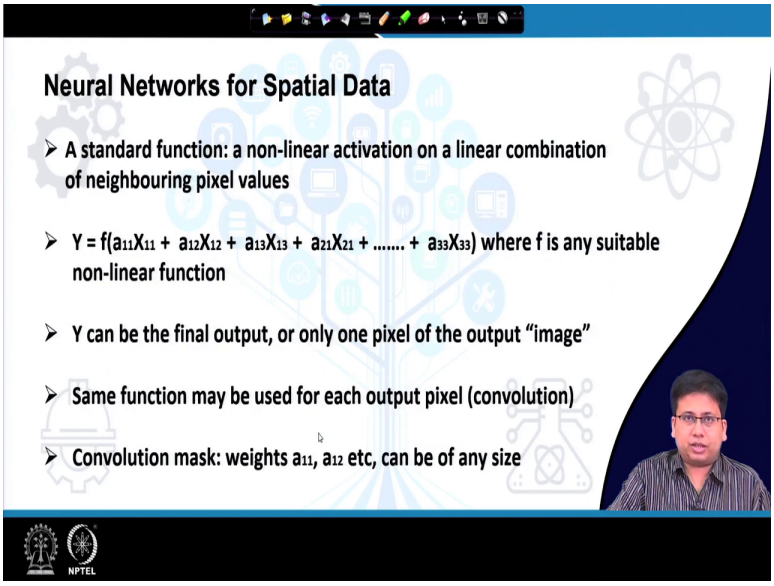
Now, it may be the same function being applied to all the pixels of the input image to calculate the output image or it can also be different functions being used being defined for the different

elements of the output image. Or, it can and also the function which f which we mentioned that function need not be like or need not take as input all the pixels of the input image, it can also focus on only one particular pixel of that input image.

Say, for example, the let us just take a trivial case the identity function. So, let us define such a function which we will just map like an image to itself. So, basically we will the function f will be such that it will like we will apply that function f separately to each pixel in the input image and the output will be just that pixel value itself. So, it will be just one to one pixel mapping, but the function f in that case its input will be just a single pixel of the input image and we will apply that function f to each input pixel separately one by one.

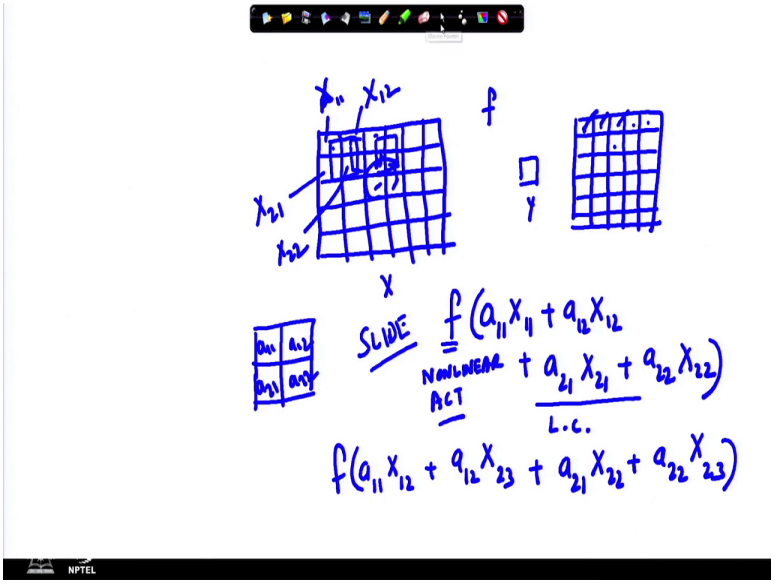
Now, the so, any whatever function it is any function can be represented by a neural network.

(Refer Slide Time: 10:40)



So, there is this famous theorem which like which claims this kind of a thing that whatever function we f we are talking about like if it is a continuous function then like we will be able to represent it by a neural network. Now, a standard function which is used in many such situations is a non-linear activation on a linear function of neighboring pixel values.

(Refer Slide Time: 11:08)



So, let us like let us take an example. So, let us say that this is the input image. So, these are the rows and these are the columns, now I want to define the function f such that let us say and let us say it is it that function f like impacts these 4 pixels and the output is some value like this. Now, these value these like it is possible that this is the only output that is this is X and this is Y that is a possibility, alternatively a possibility is that the output is another image like this.

So, in that case, like we need to define the function for like for each of these pixels in the output image. So, like either it can be different functions for each pixel or it can be the same function, but applied to different sets of input pixels for like a simple example can be that there is only a single function f which is like let us say we apply it to these 4 pixels whatever value we get we put it here.

Then again I that same function I next consider these 4 pixels and I place it here then after that I consider these 4 pixels I apply the function and I place it here and so on and so forth ok. So, now, what kind of function can this be? So, one possibility is, so let us say that like these values let us denote by a_{11} or let us say X_{11}, X_{12} similarly this one can be X_{21}, next one is X_{22} and so on.

So, one possibility is that we let us consider $f(a_{11}X_{11} + a_{12}X_{12} + a_{21}X_{21} + a_{22}X_{22})$. So, like you can say that this is a linear combination and this is f this can be any non-linear mapping, we can we sometimes call this as the activation function.

So, like this way we get some value this value we can put as the like the value of this output pixel. Now, the that same set of these a 1 a so, these a_{11}, a_{12}, a_{21}, a_{22} these like we can consider to be some constants some that is whose values may not be known, but like the function which we are talking about like we can define it in terms of. So, we can consider a_{11}, a_{12} etcetera as the parameters of that function ok.

So, now when we are trying to calculate the value of this pixel then what we will be doing is we will be using $f(a_{11}X_{12} + a_{12}X_{23} + a_{21}X_{22} + a_{22}X_{23})$. So, it is like initially we were focusing on these 4 the first 4 pixels after that I am focusing on the next 4 pixels. So, this process this is known as a convolution.

So, it is like saying that I have an image like this and I have a convolutional filter like this. So, we can say a_{11}, a_{12}, a_{21}, a_{22} and now this convolution filter what I am effectively doing is I am moving it the sliding it all over the image. So, I will; so I will like initially put it in the these this position after that is and then slide it horizontally and then like I will just move it horizontally and so on that way I am I will be like computing the output value for every of each of these output pixels along the first row of the output image.

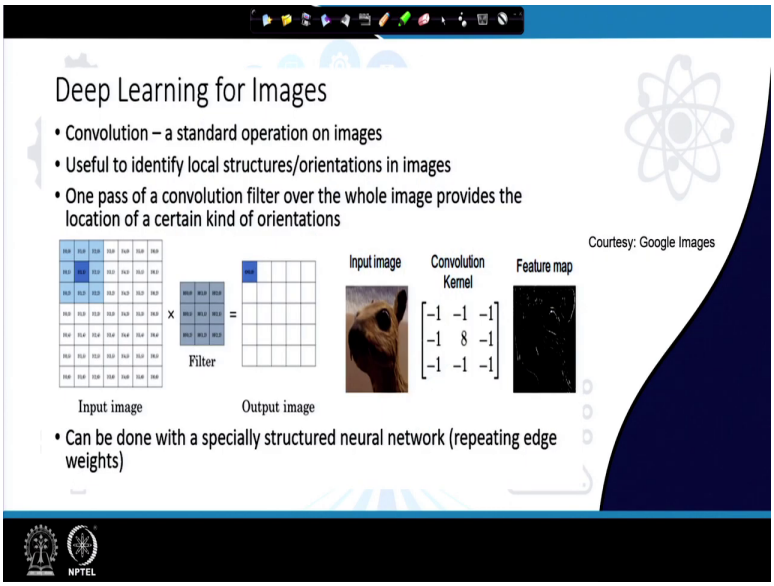
After that, I will shift this filter down and I will be focusing on these 4 pixels and then again I will slide it horizontally like this and then I will be getting all the values out of this row in the output image and then again I will move it down. So, it is like I am sliding it horizontally and then vertically, horizontally and then vertically and so on. And now when I am sliding that the filter I will I may like either I can slide it smoothly or continuously or I can take jumps that is like in the I am initially placing it in this on the first 4 pixels.

Then after that I can like from here I can jump till the next 4. So, in that case I am taking a jump of two steps or I can just slide it by one step or in I could have jumped it by like say three steps

or four steps also, let say after covering these 4 pixels now suddenly I cover these 4 pixels. So, in that case I have taken a big jump. So, this is known as the slide of the convolution operation ok.

So, this convolution operation this is an like it is an important task of the of image processing and so, these the filter which we talked about the which consists of this a 1 1 a 1 2 etcetera this is sometimes known as the convolution mask.

(Refer Slide Time: 17:38)



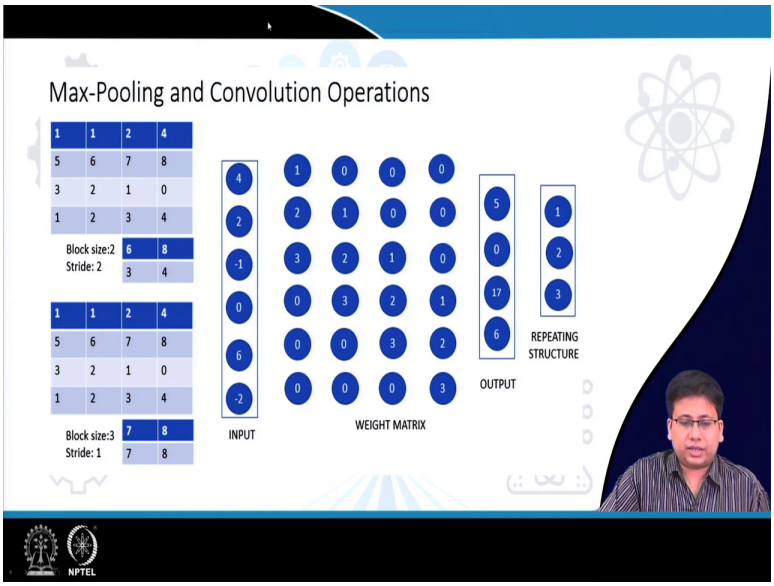
So, this is or these kind of convolutional masks they are this or the convolution operation this is often used on images for various purposes such as coarsening the image or edge detection or so on. So, like in the image processing community they often define these convolutional kernels in such a way or the convolutional masks or kernels in such a way as to like identify certain special types of orientations in the image. Say like; for there are kernels for detecting horizontal edges, kernels for detecting vertical edges and so on.

So, what I mean by that is like there are certain kernels so, if you convert this input image with that kernel like the in the response image wherever there is some kind of like say let say a horizontal edge some like for example, here you can see the face of this animal. So, like on along the outer end of this face you can understand that there is some kind of an edge. Similarly, like the like if you consider this these outlines, so, these are something like edges. So, these edge is of

course, it is neither horizontal nor vertical, but then there are like an image can have horizontal or vertical edges.

So, the idea is that the output will be some kind of a binary image or like as the as you are seeing this output. So, the like wherever the those pixels which constitute a horizontal edge those pixels will be shown in white the other that is they will have an a high output value while the other pixels they will have low output value and hence they will appear as black. So, this is how the operation of convolution is used in the domain of image processing.

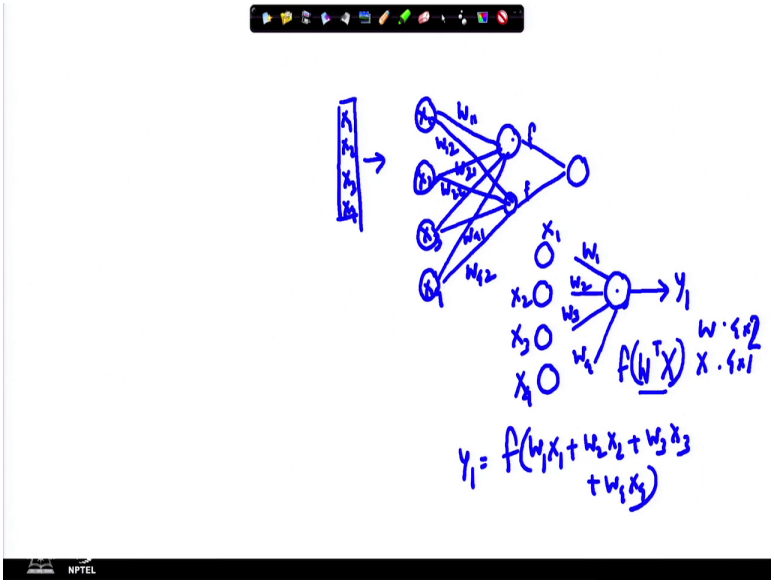
(Refer Slide Time: 19:34)



Now, this convolution operation can be implemented with the help of a neural network. So, neural network we know is capable of implement like of expressing various mathematical functions, now this convolution is a special kind of mathematical function which is capable like which is very useful in image processing and signal processing. And it has been shown that this convolution function can also be implemented using a neural network.

So, how is that? So, the way a neural a standard neural network operates is as follows. So, what a neural network usually does is it calculates functions like this.

(Refer Slide Time: 20:24)



So, let us say we have this kind of a thing, an input vector let us say

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}.$$

So, the neural network basically considers this vector as like input nodes which contain the values X_1, X_2, X_3, X_4. Now, what the neural network does is like it has some hidden units let say there are 2 hidden units let say for simplicity and then there are edges like this which connect the input nodes with the hidden nodes. And then from the hidden nodes there let us say there is an output node also.

So, what happens in each node is that so, like any node of the neural network what it does is, it receives inputs from other nodes in the previous layer. Each of these inputs has some kind of a weight let us say. So, these weights maybe we can call them as W_{11}, this one maybe I can call it as W_{12}, this one is W_{21}, this one is W_{22}. So, similarly this one is W_{42}, this one will be W_{41} and so on and so forth.

So, like here let us say that in any neuron in the except the input neurons in any hidden layer let us say that this is a neuron which has 4 connections from the previous layer and their weights are W_1, W_2, W_3, W_4. And let us say the corresponding nodes in the previous layer they contain the values X_1, X_2, X_3, X_4, correct?

So, the value which the neuron will calculate in this case is $Y_1 = f(W_1X_1 + W_2X_2 + W_3X_3 + W_4X_4)$. So, similar so, like what it is doing is basically something like a matrix operation. So, you can imagine these X_1, X_2 these to be a matrix X and these W to be or rather a vector X and these $w's$ they are like another vector W. So, we are effectively calculating the dot products between these two things like we are getting $f(W^TX)$.

So, similarly if you like if there are multiple if instead of considering only one neuron like this, if you consider two neurons then we can you can say that like the different weights we like we can consider them to be different columns separate columns of a matrix. And then like instead of defining a row I mean a vector operation like this we can consider this as a matrix multiplication.

So, where in this case W is a $4X1$ matrix and X is also a $4X1$ matrix. Now if there are like if you are considering 2 hidden nodes then we have 2 sets of weights. So, it will be we can we make it as a $4X2$ matrix. So, then this W^TX this will have size $2X1$ and on that we can each of them I can like apply this function f. So, this can act on this function f such a function which acts on every element of the vector. So, that it is like saying that on each of these outputs we are applying that same activation function f. So, that is how our neural network usually works.

Now the same thing can be done in a convolutional I mean in a the same thing can be done to replicate the simulate the operation of convolution. So, let us say that this is the like an input vector and now so, if you do the multiplication with this kind of a weight matrix then you can just convince yourself that this is going to be the output. But the interesting thing to note is this output is the same as what you get if you carry out the convolution operation on this input using this kind of a this convolution kernel.

So, just to demonstrate once so, $1, 2, 3$ these are the like values. So, initially when you are doing the convolution. So, initially you place it against the first three elements. So, $4X1 = 4$,

$2X2 = 4, -1X3 = -3$. So, $4 + 4 - 3 = 5$. So, the answer like so, here the first output is 5 and you can check here the indeed the first output is 5.

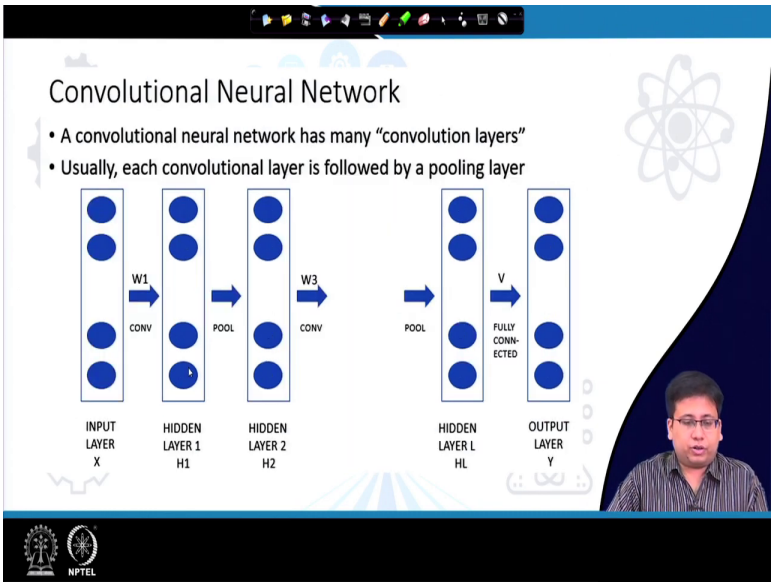
Next, you slide that this kernel or mask to cover the next three inputs. So, here $2X1 = 2$, $-1X2 = -2$ and $0X3 = 0$. So, the result is $2 - 2 + 0 = 0$ and you see 0 in the output. So, similarly if you slide it to the next 3 so, then $-1X1 = -1$, $0X2 = 0$ and $6X3 = 18$. So, $-1 + 0 + 18 = 17$ and you have 17 here and so on.

So, here you see that if you do the convolution of this input with this convolution mask then this is output you will get, but if you do the matrix multiplication with these kind of a weight matrix which has this $1, 2, 3$ this mask placed as a repeating structure in the different columns then also you will get the same output. So; that means, the convolution operation can actually be expressed as a matrix operation provided your matrix is defined in that particular way.

So, what happens in a convolutional neural network is that the different like a edge weights which we talked about are defined in such a way that it simulates the convolution operation. And now along with convolution another operation that is very important in the domain of image processing and computer vision that is max-pooling. So, let us say these are your different pixel values. So, what you do is you divide these into blocks and in each block you store only the largest value that is the max pooling.

So, like the block size can be different like in this case the block sizes are 2. So, like your you consider a block size 2 like this. So, here of course, the largest value is 6. So, you store it next you take a stride. So, the in the previous in case of convolution kernel also I mentioned this property of stride. So, we can place the kernel in the here. So, in that case you will get you cover these 4 pixels and here the largest value is 8 and so on. So, similarly you can use the like stride value also.

(Refer Slide Time: 28:06)

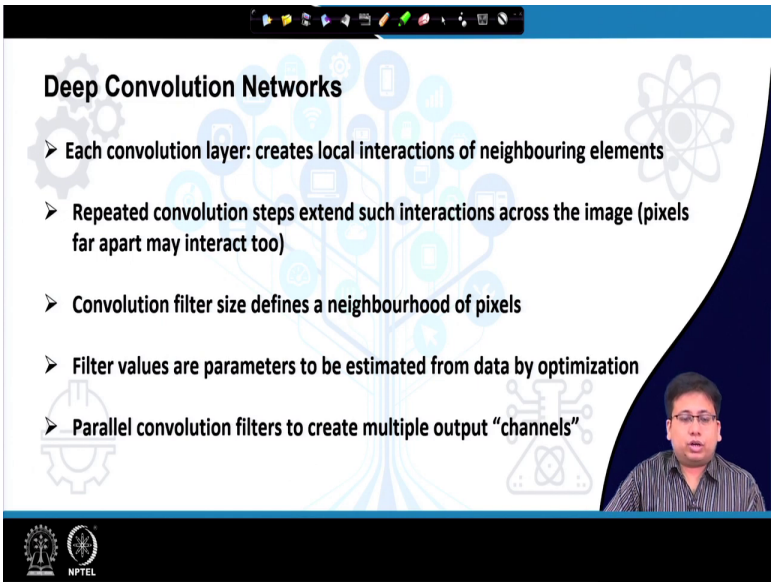


So, basically this is in a typical convolutional neural network this is what the network looks like. So, there are multiple layers like this. So, usually the of course, first is the input layer, after that there is one convolutional layer. So, these W_1 the weights are arranged in such a way that this like this kind of repeated structure is used, the apart from the repeated structure the remaining elements are all 0 that is that is it is a sparse matrix.

Now, this weight matrix is sparse, but it has these repeated structure. So, that the operation it performs is effectively a convolution and similarly just like convolution this pooling operation can also be expressed as a matrix operation. So, the next layer is a pooling layer and then again we have a convolutional layer again we have a pooling layer and so, this way it operates and finally, we get a fully connected layer and that gives us the output.

So, this is what our standard convolutional neural network looks like. So, here I have written drawn the input and the output in the form of what may be a vector, but in general the input and output can be images also.

(Refer Slide Time: 29:19)

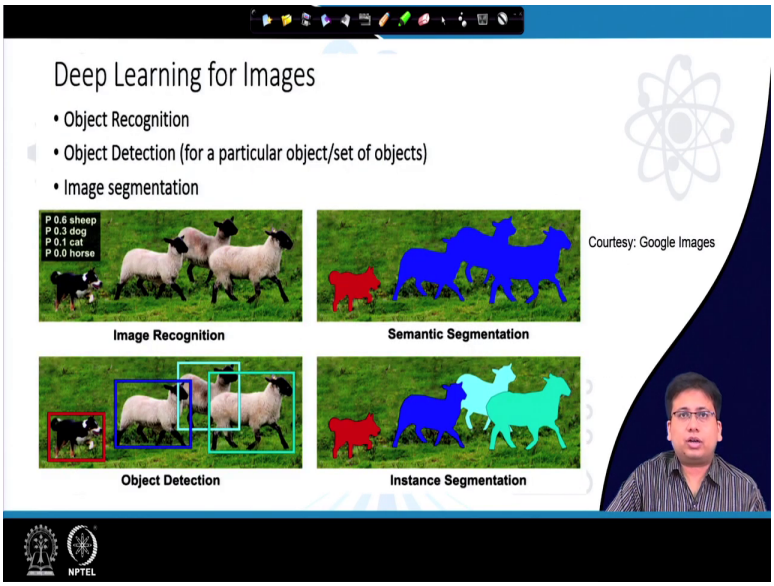


So, a deep in a deep convolutional network each convolutional layer creates local interactions of the neighboring elements and repeated convolution steps extend such interactions across the image and so that the pixels far apart can also interact and convolution filter size that basically defines a neighborhood of in pixels. So, if the size is say $2X2$; that means, like $2X2$ like a $2X2$ neighborhood of pixels they will interact to produce the output.

If it is $3X3$ then like a 9 neighboring pixels like a will interact to produce the output values and so on. So, these convolutional like these are usually the user defined parameters that is the size of the filters and so on, but the values of the parameters I mean the values of the filter the those a_{11}, a_{12} etcetera which I mentioned they are usually estimated from the data by optimization.

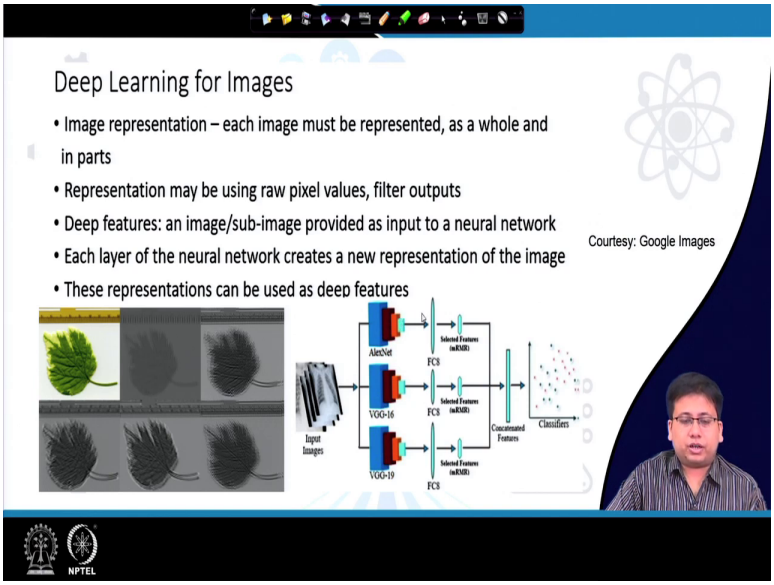
And we can like we at the same layer we can actually have parallel convolutional filters. So, the output need not even be a single image it can be a like it can be a tensor where like there are it is a multi-channel image just like an input.

(Refer Slide Time: 30:36)



So, this convolutional neural network this is used for various tasks in computer vision such as this image recognition, object detection, image segmentation and so on.

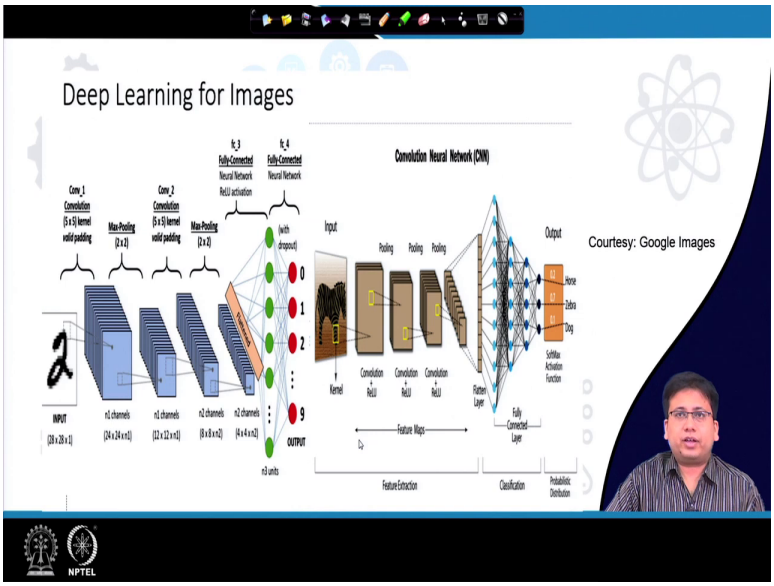
(Refer Slide Time: 30:50)



Like now in case of computer vision, one of the important tasks is feature representation of the images that is whenever I have any image I want a mathematical representation of the image

through vectors or something like that. So, earlier the computer vision or image processing scientists used to come up with their own features borrowing from signal processing which are typically various filters. So, those pixels those features used to indicate say something like texture, edges, etcetera.

(Refer Slide Time: 31:22)

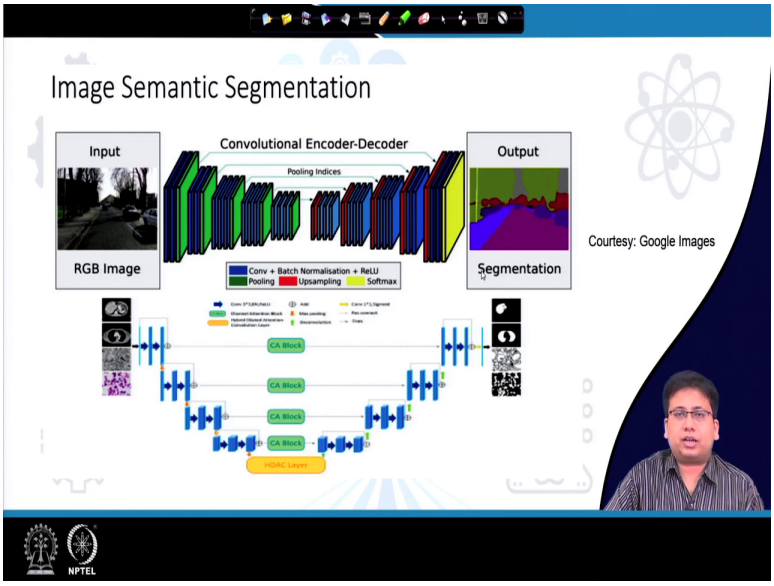


But now the neural network itself calculates the different representations of the image which are like which it themselves can serve as the features. So, like whenever we have one layer of the this neural network it creates a different representation of the image and after like doing the convolution operation and so that output that intermediate outputs which we get those are known as the feature maps. And they may themselves be useful for further classification of the or of further operations on the image.

(Refer Slide Time: 31:50)

So, we the need of defining features has gone away. So, this object detection is another very important problem which is solved with the help of convolutional neural network where we are searching for specific objects in an image. So, we it the that object can be present anywhere and in any size so we like we and simultaneously search at different locations and at different scales using this convolutional neural network.

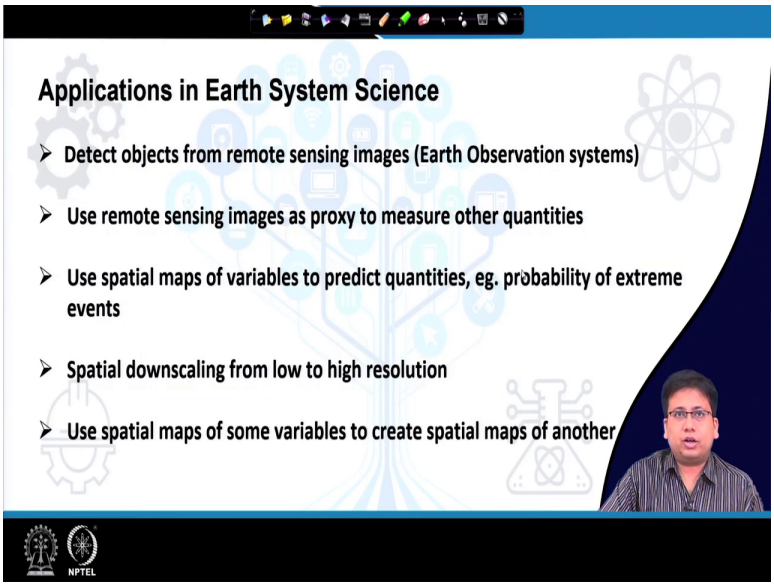
(Refer Slide Time: 32:21)



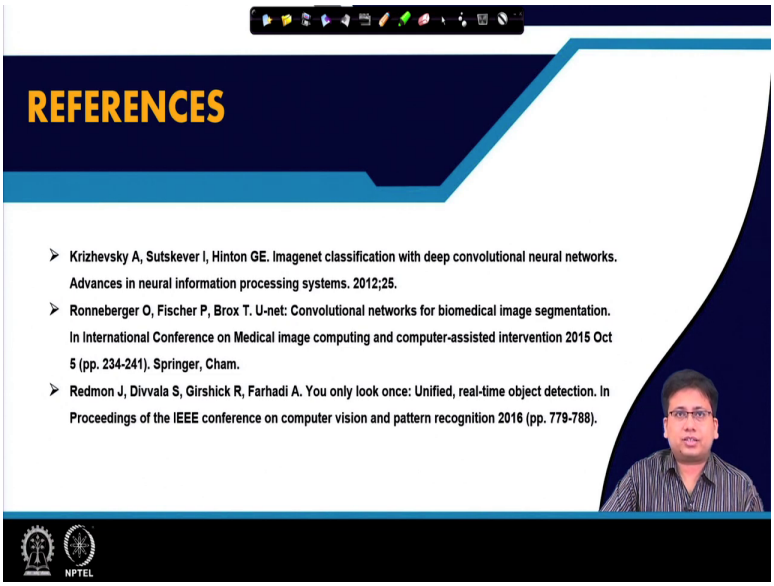
So, basically we are like solving a classification problem at different locations and scales. And another important task is that of semantic segmentation, where we have an image like this and we the output is some kind of a segmented image where like or with which we can call it as a smoothed image. And this is achieved with the help of another special type of neural network called the U - Net which first like encodes the image to some kind of a coarse form and then or I mean creates a small representation of the image.

And then brings it back to the original size, but in the process it has thrown away the various details and giving only a coarse grain representation.

(Refer Slide Time: 33:02)

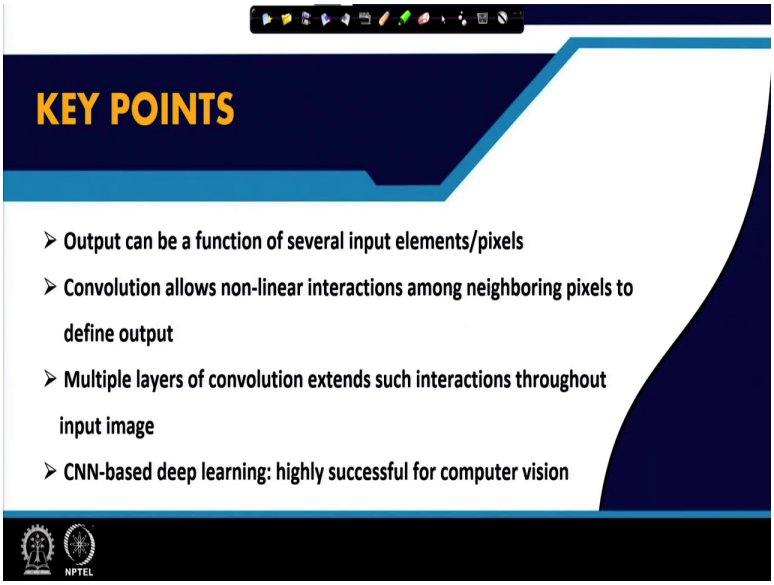


(Refer Slide Time: 33:14)



Now, this like these methods they have various applications in earth system sciences. So, we will discuss these applications in great detail later and so, these are a set of references where we I have basically provided you links to the different papers of computer vision like so that you can understand these different models.

(Refer Slide Time: 33:24)



And so, these are the key points. So, the outputs can be of an the of these operations they can be a function of several input elements or pixels, the this convolution operation this allows non-linear interactions among the neighboring pixels to define the output. Now, we can have multiple layers of convolution to extend such interactions throughout the input image. And finally, this CNN - based deep learning, this is highly successful for computer vision and also has applications in earth system science. So, with that we come to the end of this lecture we will continue this discussions further till then goodbye.