

**Neural Networks for Signal Processing-I**  
**Prof. Shayan Srinivasa Garani**  
**Department of Electronic System Engineering**  
**Indian Institute of Science, Bengaluru**

**Lecture – 64**  
**Demo – Motivation for CNN**

Hello everyone, let's explore why convolutional neural networks (CNNs) are preferred over multilayer perceptrons (MLPs) for tasks like image classification. In an image classification problem, we are given a set of images and need to classify them into predefined categories.

(Refer Slide Time: 00:59)



The image shows a YouTube video player interface. The video title is "ee53 lec64 Demo – Motivation for CNN" and the video title is "Motivation". The video content shows a slide titled "Image Classification : Main problem for Computer Vision". The slide features an image of a cat and text: "Suppose we had a set of classes : cat, truck, boat." with an arrow pointing to the word "Cat". The video player shows a progress bar at 0:59 / 11:06 and various control icons.

For instance, consider categories such as cat, truck, and boat. When a human is shown an image, we can instantly identify it as a cat. However, when the same image is fed into a

computer, it is interpreted as a 3D array of numbers, with each integer ranging from 0 to 255.

The 3D aspect comes from the RGB channels of the image, each pixel contains three values corresponding to the red, green, and blue components. The computer must then determine which class these numerical values correspond to, a task that is not straightforward because recognizing objects from just numbers is challenging. Therefore, our goal is to build a classifier that can accurately assign the correct label to each image.

For this purpose, let's consider the CIFAR-10 dataset, which comprises 10 classes: plane, car, bird, cat, deer, dog, frog, horse, ship, and truck. The dataset includes 50,000 training images and 10,000 testing images, with each image sized 32x32 pixels.

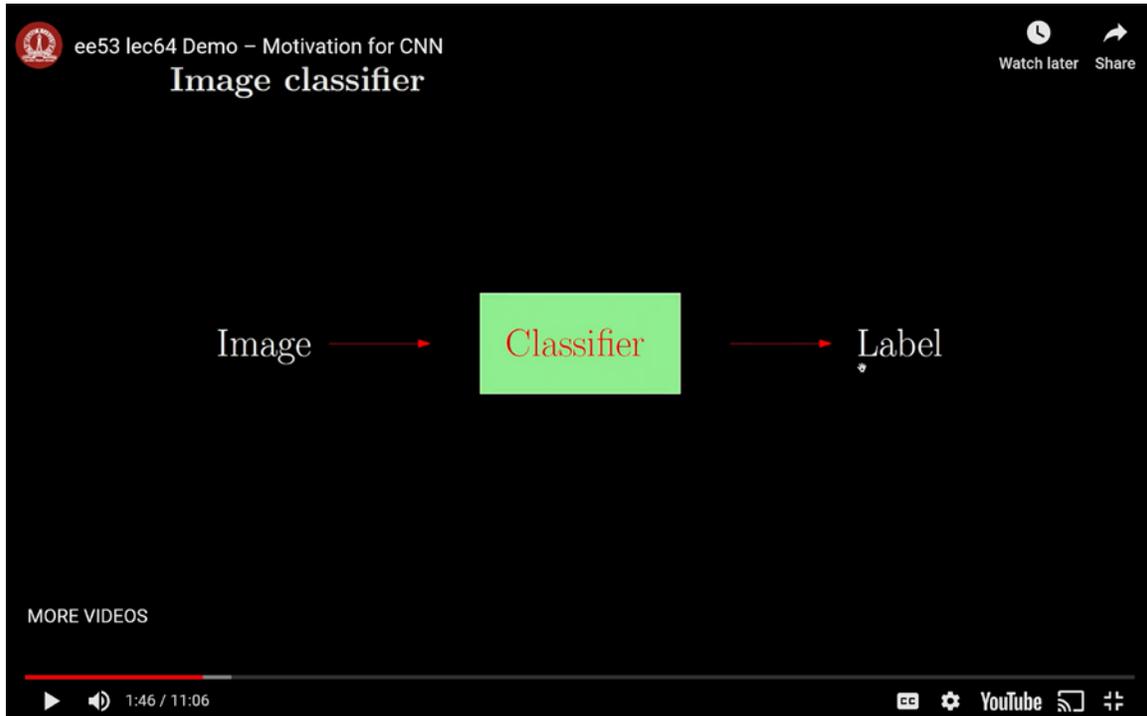
(Refer Slide Time: 01:36)

The screenshot shows a YouTube video player with a slide titled "Problem: The semantic gap". The slide content includes the text "Images are represented as 3D arrays of numbers, with integers between [0,255]". Below this text is a photograph of a white and orange cat lying down. A red box highlights a small portion of the cat's face, and a red-bordered grid of 32x32 numerical values is shown to the right, representing the pixel data of that region. The video player interface shows the video is at 1:36 / 11:06.

Let's start by examining how a multilayer perceptron (MLP) would approach classifying these images. The general architecture of an MLP includes an input layer and an output layer, with a single hidden layer for simplicity in this example. The input size is 32x32x3,

given the RGB channels, and the output layer has 10 neurons, corresponding to the 10 classes.

(Refer Slide Time: 01:46)



The process of training an MLP involves the following steps:

- 1. Forward Propagation:** Feed the images into the network, obtain predicted labels, and compare them with the true labels to compute the loss function.
- 2. Backpropagation:** Calculate the gradients of the loss function with respect to the network's parameters.
- 3. Parameter Update:** Adjust the network parameters using the calculated gradients.

These steps are repeated for all images in the training dataset, after which the network's performance is evaluated on the test images. Testing helps assess the accuracy of the network in classifying images correctly.

Next, let's discuss the impact of hidden neurons on classification accuracy.

(Refer Slide Time: 02:13)

ee53 lec64 Demo – Motivation for CNN

## CIFAR 10 dataset

- 10 classes
- 50,000 training images
- 10,000 testing images
- 32 × 32 image size

Plane	Car	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck

MORE VIDEOS

2:13 / 11:06

YouTube

(Refer Slide Time: 02:56)

ee53 lec64 Demo – Motivation for CNN

## Multilayer Perceptron

Input layer

Hidden layer

Output layer

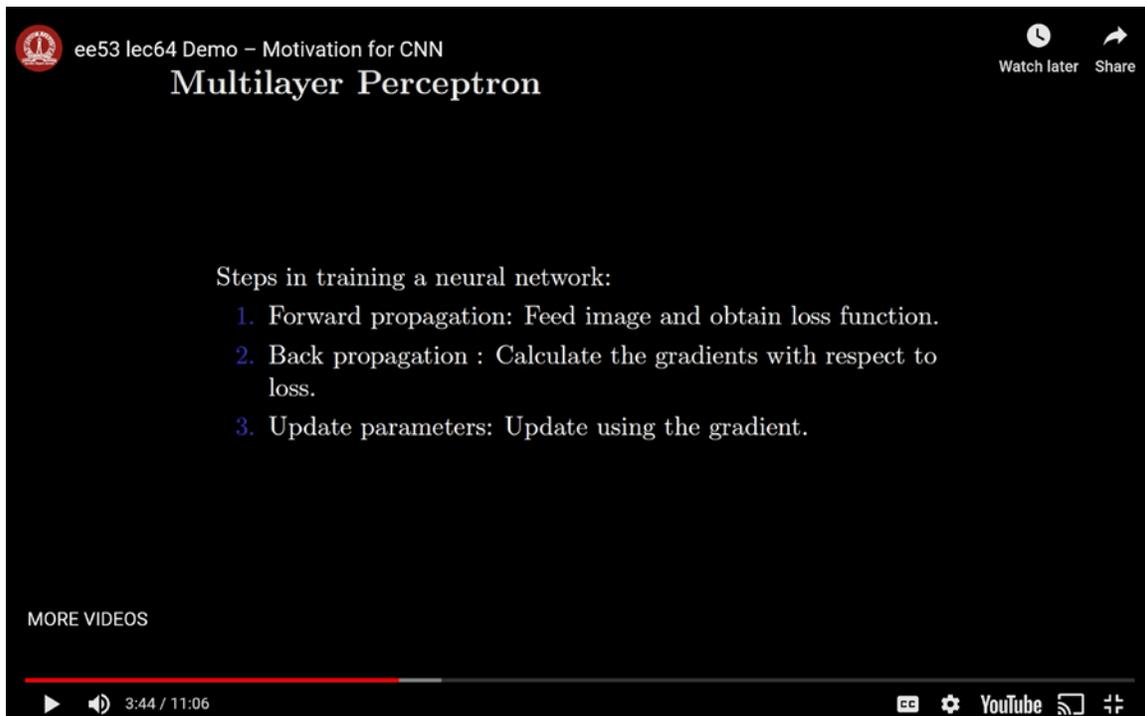
MORE VIDEOS

2:56 / 11:06

YouTube

We observe that with 10 neurons in the hidden layer, the accuracy achieved is 40.6%. Increasing the number of neurons to 50 improves the accuracy to 48%, and with 100 neurons, it rises to 51.8%. However, beyond this point, further increases in the number of hidden neurons do not significantly enhance the accuracy. This phenomenon is attributed to overfitting during training. Let's delve deeper into this issue.

(Refer Slide Time: 03:44)



The image shows a YouTube video player interface. At the top left, there is a red circular logo with a white figure, followed by the text "ee53 lec64 Demo – Motivation for CNN". The main title of the video is "Multilayer Perceptron" in a large, white, serif font. In the top right corner, there are icons for "Watch later" and "Share". The main content of the slide is a list of "Steps in training a neural network:" followed by three numbered items: 1. Forward propagation: Feed image and obtain loss function. 2. Back propagation : Calculate the gradients with respect to loss. 3. Update parameters: Update using the gradient. At the bottom left, there is a "MORE VIDEOS" link. The video player controls at the bottom show a play button, a volume icon, and a progress bar indicating 3:44 / 11:06. There are also icons for closed captions, settings, YouTube logo, and a full screen icon.

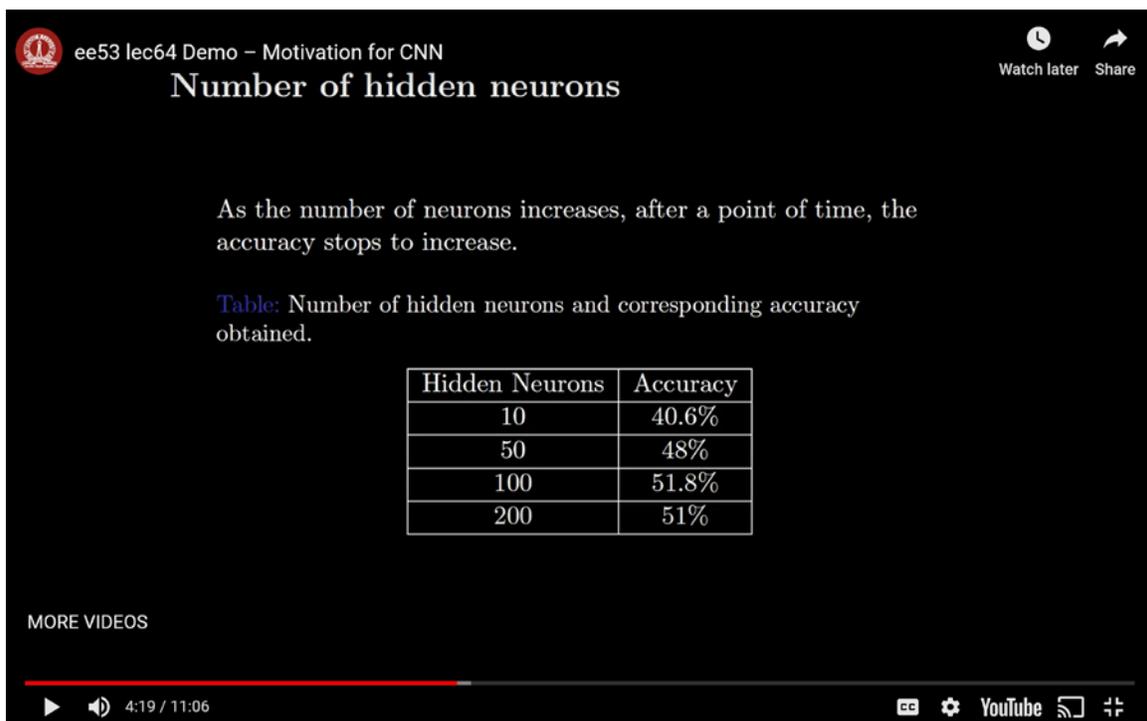
Consider a network with 100 hidden neurons. The image size is  $32 \times 32 \times 3$ , which equals 3,072 pixels. Consequently, the number of parameters between the input layer and the hidden layer is 307,200, a substantial number, given that we only have 50,000 training samples. If the image size were increased to  $960 \times 70 \times 3$ , converting the image into a single vector for the multilayer perceptron would result in an enormous input layer and an excessively large number of parameters. This would undoubtedly lead to overfitting and increased complexity of the network, thereby prolonging the training time.

Despite increasing the number of hidden neurons, the multilayer perceptron does not yield better accuracy. Is there a better approach? According to the experiments conducted by

Hubel and Wiesel on the visual cortex of cats, features are organized hierarchically into simple cells, complex cells, and hyper-complex cells, which identify low-level, mid-level, and high-level features, respectively. This hierarchical concept has been incorporated into neural networks for image classification.

The fundamental components of a convolutional neural network (CNN) are convolution and pooling. During convolution, we identify features by using a filter or kernel to extract relevant information from the input. Pooling introduces invariance by discarding some values, which we will explore further.

(Refer Slide Time: 04:19)



The image shows a YouTube video player interface. At the top left, there is a red circular logo with a white 'e' and the text 'ee53 lec64 Demo - Motivation for CNN'. To the right of the logo is the title 'Number of hidden neurons'. In the top right corner, there are icons for 'Watch later' and 'Share'. The main content of the slide is a text block that reads: 'As the number of neurons increases, after a point of time, the accuracy stops to increase.' Below this text is a caption: 'Table: Number of hidden neurons and corresponding accuracy obtained.' Underneath the caption is a table with two columns: 'Hidden Neurons' and 'Accuracy'. The table contains four rows of data: (10, 40.6%), (50, 48%), (100, 51.8%), and (200, 51%). At the bottom of the slide, there is a 'MORE VIDEOS' link. The video player interface at the bottom shows a play button, a volume icon, the time '4:19 / 11:06', and icons for 'YouTube', 'Settings', and 'Full Screen'.

ee53 lec64 Demo - Motivation for CNN

## Number of hidden neurons

As the number of neurons increases, after a point of time, the accuracy stops to increase.

Table: Number of hidden neurons and corresponding accuracy obtained.

Hidden Neurons	Accuracy
10	40.6%
50	48%
100	51.8%
200	51%

MORE VIDEOS

4:19 / 11:06

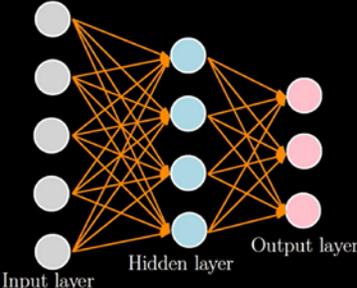
YouTube

In convolution, given a  $32 \times 32 \times 3$  image, consider a filter of size  $5 \times 5 \times 3$ . This filter is moved across the image from the top-left to the bottom-right corner. At each position, the dot product of the filter and the corresponding image values is calculated to produce a single output value. This process results in an activation map. For a  $32 \times 32 \times 3$  image convolved with a  $5 \times 5 \times 3$  filter, the activation map size is  $28 \times 28 \times 1$ . If we use 6 such filters, the output will be a  $28 \times 28 \times 6$  activation map.

(Refer Slide Time: 05:29)

ee53 lec64 Demo – Motivation for CNN

## Neural Network



Input layer Hidden layer Output layer

Say 100 hidden neurons.

- ▶ Size of image =  $32 \times 32 \times 3 = 3072$ .  
Number of parameters = 3,07,200.
- ▶ Size of image =  $960 \times 720 \times 3 = 20,73,600$ .  
Number of parameters = 20,73,60,000.

MORE VIDEOS

5:29 / 11:06

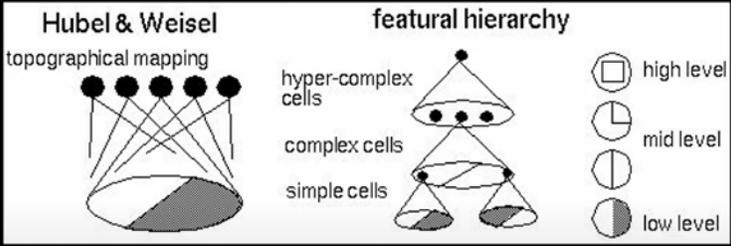
YouTube

(Refer Slide Time: 05:54)

ee53 lec64 Demo – Motivation for CNN

## Visual Cortex

**Hubel and Weisel:** Feature identification using low, mid and high level features.<sup>1</sup>



Hubel & Weisel topographical mapping

featural hierarchy

- hyper-complex cells
- complex cells
- simple cells

- high level
- mid level
- low level

MORE VIDEOS

<sup>1</sup>Hubel, David H., and Torsten N. Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex." *The Journal of physiology* 160.1 (1962): 106-154.

5:54 / 11:06

YouTube

This activation map becomes the new input with a size of  $28 \times 28 \times 6$ . In a CNN architecture, multiple convolution layers are stacked sequentially for feature extraction. For instance, with six  $5 \times 5 \times 3$  filters, the output is  $28 \times 28 \times 6$ . In the subsequent layer, if we use ten  $5 \times 5 \times 6$  filters, the output size becomes  $24 \times 24 \times 10$ , and so forth.

The key point to highlight here is that the only parameters needing to be learned during training are the filters, specifically the  $5 \times 5 \times 3$  and  $5 \times 5 \times 6$  filters. While the complexity of the network does increase with the number of filters, it remains significantly lower than that of a multilayer perceptron.

Next, let's discuss the pooling layer, using max pooling as an example. Max pooling reduces the number of activations by 75%. This process involves down-sampling the input by a factor of 2 along both the width and height. We achieve this by applying pooling over a  $2 \times 2$  grid with a stride of 2. For each  $2 \times 2$  grid, we identify and retain only the maximum value while discarding the remaining values.

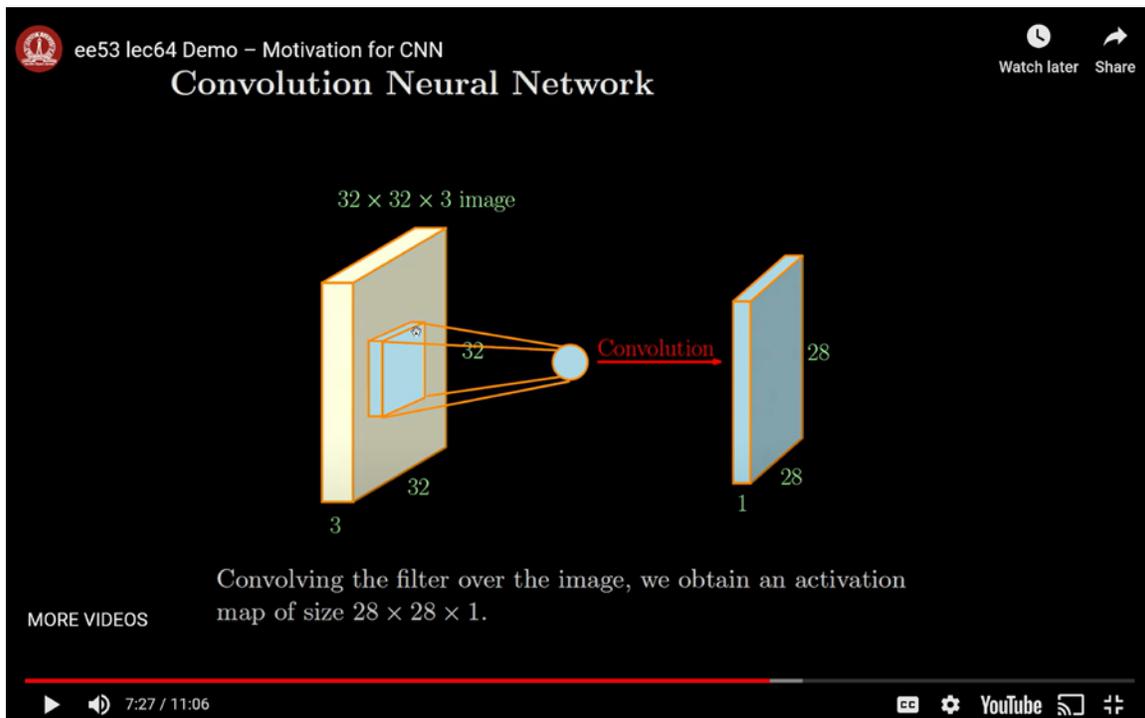
(Refer Slide Time: 06:31)

The image shows a YouTube video player interface. At the top left, there is a red circular logo and the text "ee53 lec64 Demo - Motivation for CNN". The main title of the video is "Convolution Neural Network". In the center, there is a diagram illustrating a convolution operation. On the left, a large yellow 3D rectangular prism represents the input image, labeled "32 x 32 x 3 image". The dimensions are indicated as 32 on the top and bottom edges, and 3 on the depth edge. On the right, a smaller blue 3D rectangular prism represents the filter, labeled "5 x 5 x 3 filter". The dimensions are indicated as 5 on the top and bottom edges, and 3 on the depth edge. At the bottom left of the video player, there is a "MORE VIDEOS" link. At the bottom center, there is a progress bar showing "6:31 / 11:06". At the bottom right, there are icons for YouTube, settings, and other video controls.

For instance, given an activation map, we consider a 2x2 grid, select the maximum value within that grid, and discard the other values at that location. We then move to the next 2x2 grid and repeat the process. This approach introduces a degree of invariance in the image. For example, if the image contains a cat, the cat's ear might appear in different positions across different images. Max pooling ensures that as long as the presence of an ear is detected, the exact location, whether at the top corner or the bottom, does not affect the network's performance.

By incorporating such invariance, max pooling also helps reduce the total number of parameters required for training the network.

(Refer Slide Time: 07:27)



The image shows a YouTube video player interface. The video title is "ee53 lec64 Demo - Motivation for CNN" and the slide title is "Convolution Neural Network". The slide content includes a diagram of a convolution operation. On the left, a 3D volume represents a  $32 \times 32 \times 3$  image. A  $3 \times 3$  filter is applied to a  $32 \times 32$  region of the image. An arrow labeled "Convolution" points to a resulting 3D volume representing an activation map of size  $28 \times 28 \times 1$ . Below the diagram, the text reads: "Convolving the filter over the image, we obtain an activation map of size  $28 \times 28 \times 1$ ." The video player shows the video is at 7:27 / 11:06.

The overall architecture of a convolutional neural network (CNN) typically involves feeding the input through a series of convolutional layers, followed by activation maps, and additional convolution and pooling layers as needed for the specific task. The final layers consist of fully connected layers that generate the labels for the given dataset.

(Refer Slide Time: 08:34)

ee53 lec64 Demo – Motivation for CNN

## Convolution Neural Network

32  
32  
3

Convolution  
ReLU  
 $5 \times 5 \times 3$   
filter

28  
28  
6

Convolution  
ReLU  
 $5 \times 5 \times 6$   
filter

24  
24  
10

Depth of filters increases as the layers increase.

MORE VIDEOS

Subtitles/closed captions (c)

8:34 / 11:06

YouTube

(Refer Slide Time: 09:57)

ee53 lec64 Demo – Motivation for CNN

## Pooling

MAX POOLING - Downsamples the input by 2 along the width and height by performing pooling over  $2 \times 2$  with a stride of 2. Discards 75% of the activations.

9	1	0	2
4	7	6	4
3	5	3	4
7	2	1	3

→

9	6
7	4

MORE VIDEOS

9:57 / 11:06

YouTube

This structure allows for a reduction in the number of parameters needed for training, even with an increased number of layers.

(Refer Slide Time: 10:36)

The screenshot shows a video player interface for a video titled "Convolution Neural Network" from a series "ee53 lec64 Demo - Motivation for CNN". The video content displays a diagram of a CNN architecture. On the left, an input image of a white car is shown. This image is processed through three stages of convolutional layers (CONV), each followed by a pooling layer (POOL) and two ReLU activation functions (RELU). The final output is a classification list with "car" highlighted in green, along with other classes: truck, airplane, ship, and horse. The video player shows a progress bar at 10:36 / 11:06 and includes standard YouTube controls like play, volume, and settings.

We encourage you to use the CIFAR-10 dataset to compare the performance of multilayer perceptrons with convolutional neural networks and assess the accuracy achieved. Once you're comfortable with both approaches, you can explore other datasets to further evaluate and refine your model's performance. Thank you.