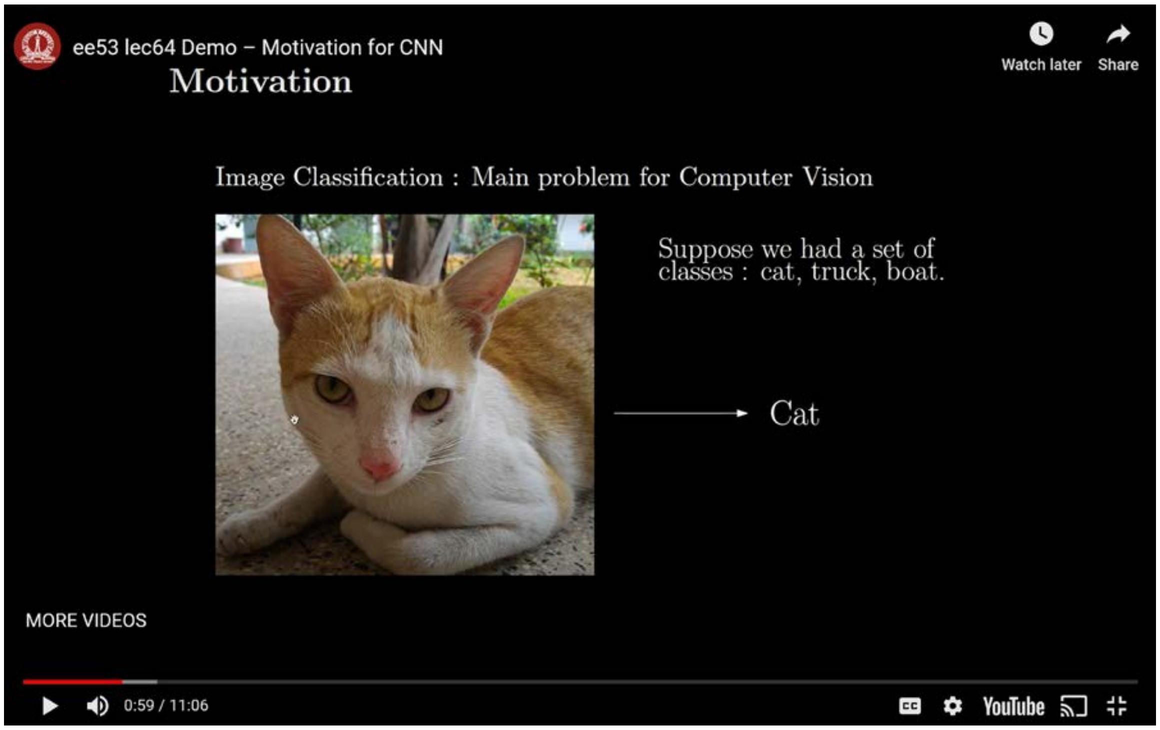
Neural Networks for Signal Processing-I

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)



Motivation

Image Classification : Main problem for Computer Vision

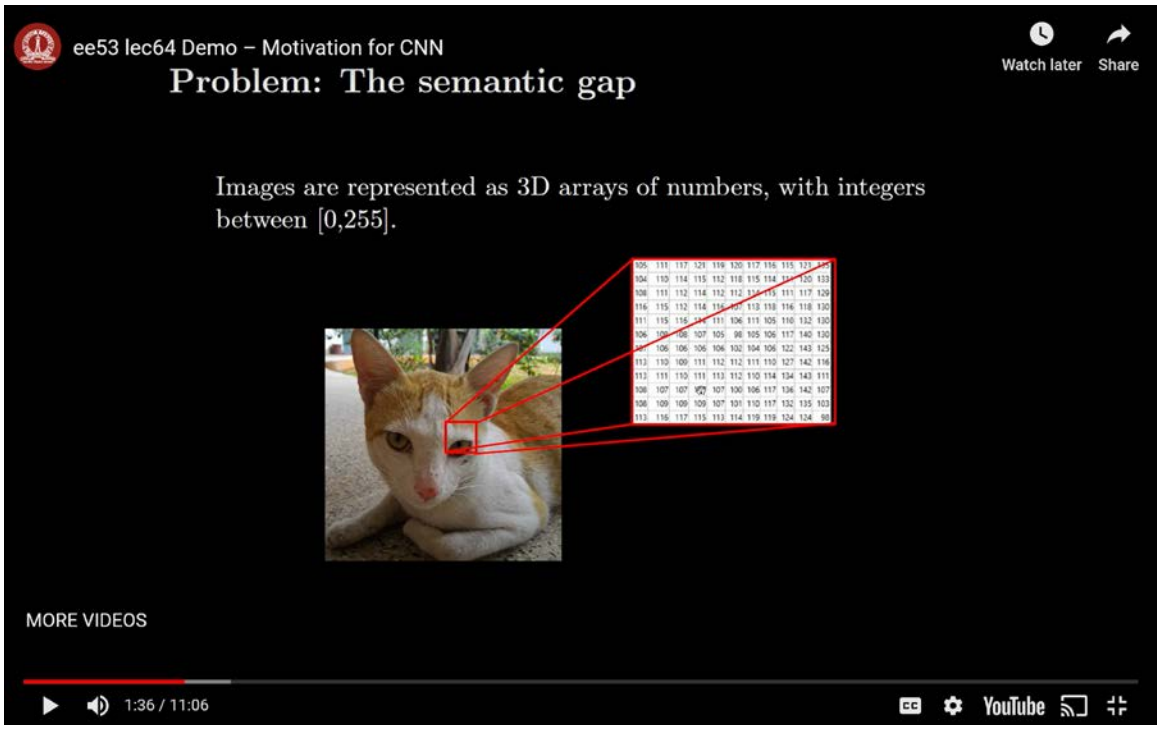
Suppose we had a set of classes : cat, truck, boat.

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)



Problem: The semantic gap

Images are represented as 3D arrays of numbers, with integers between [0,255].

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)

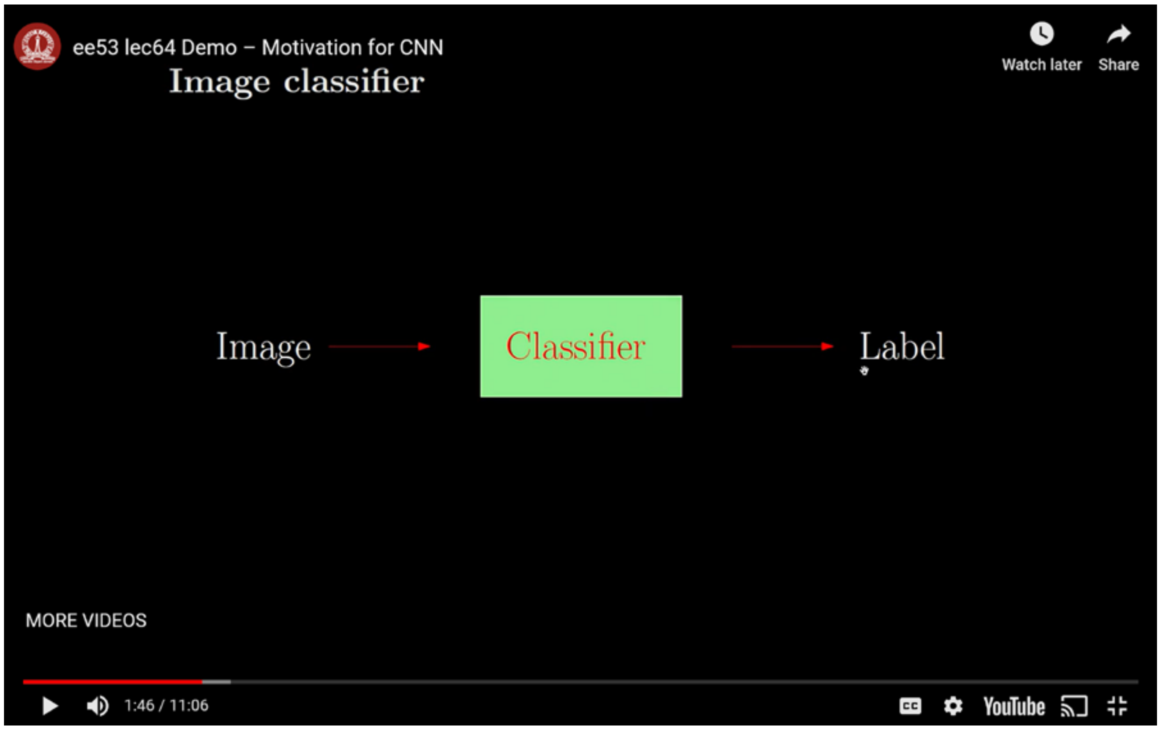


Image classifier

$$\begin{tikzcd}\text{Image} \arrow[r] & \boxed{\text{Classifier}} \arrow[r] & \text{Label}\end{tikzcd}$$

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.

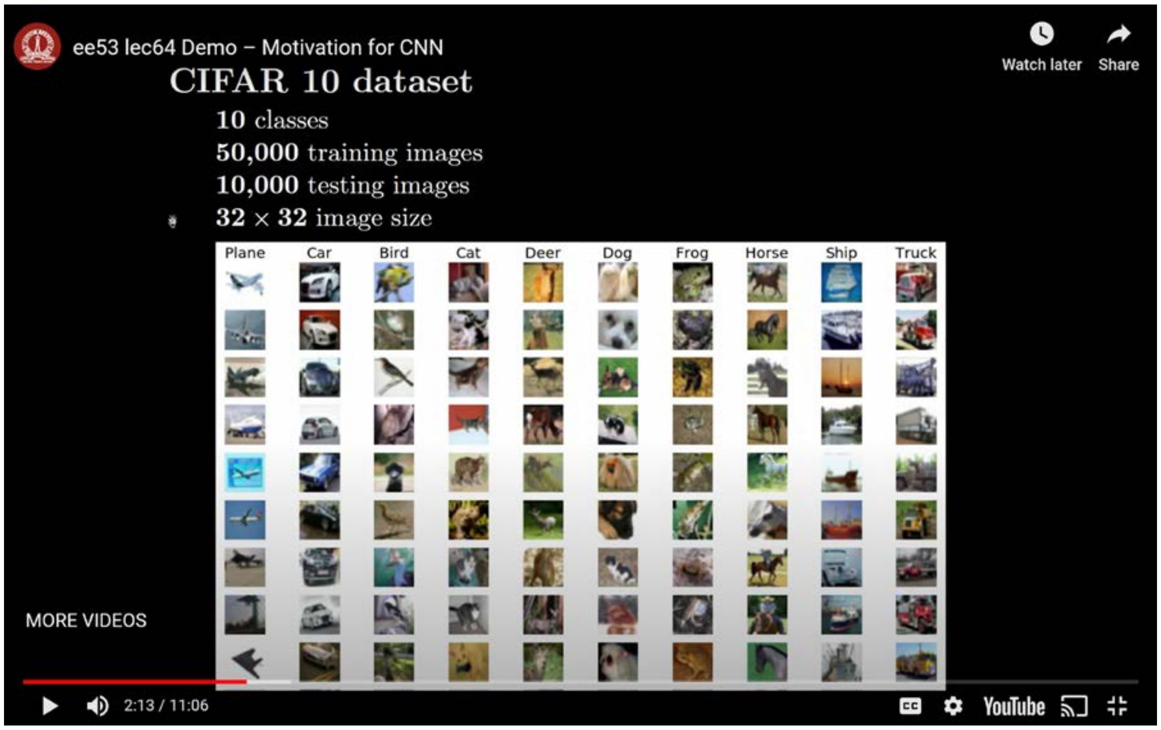
2. Backpropagation: Calculate the gradients of the loss function with respect to the network's parameters.

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.

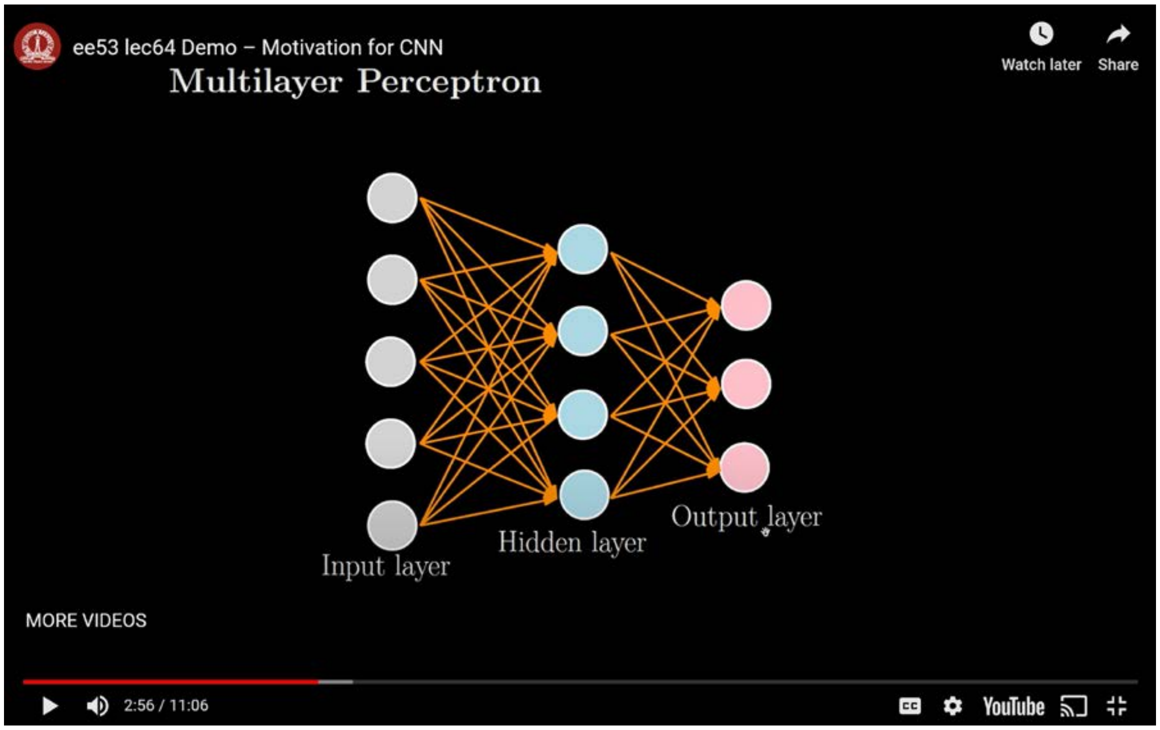
(Refer Slide Time: 02:13)



CIFAR 10 dataset

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

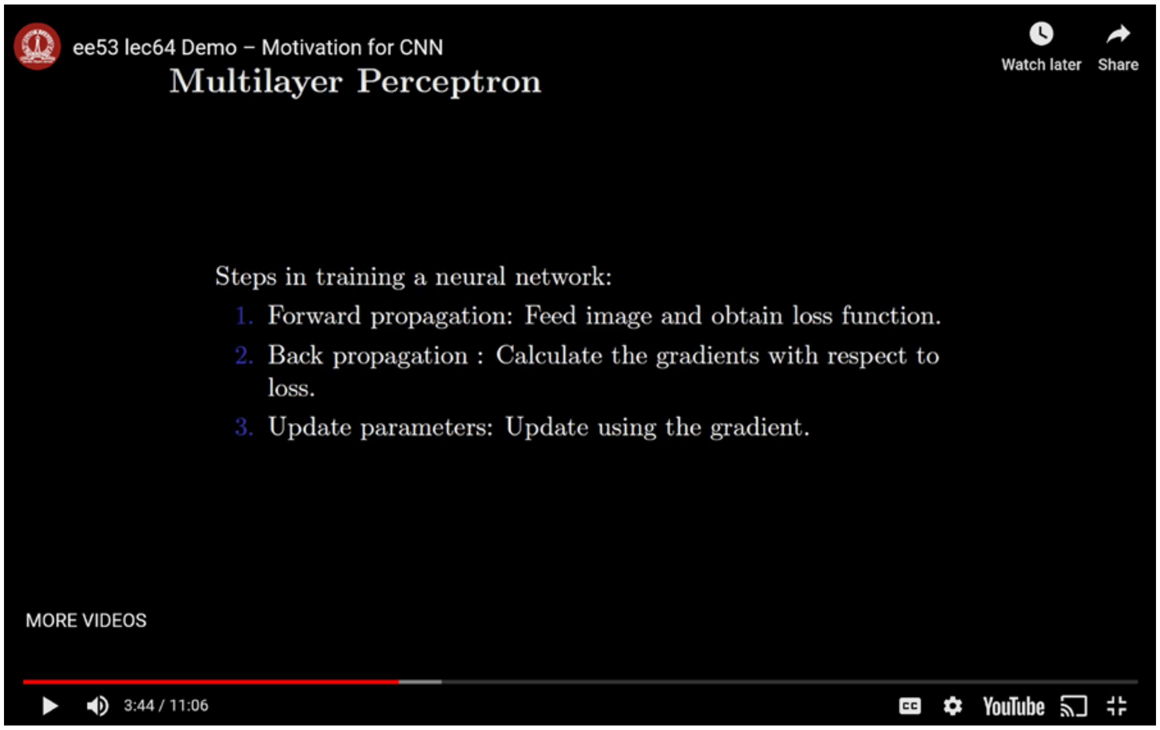
(Refer Slide Time: 02:56)



Multilayer Perceptron

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)



Multilayer Perceptron

Steps in training a neural network:

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.

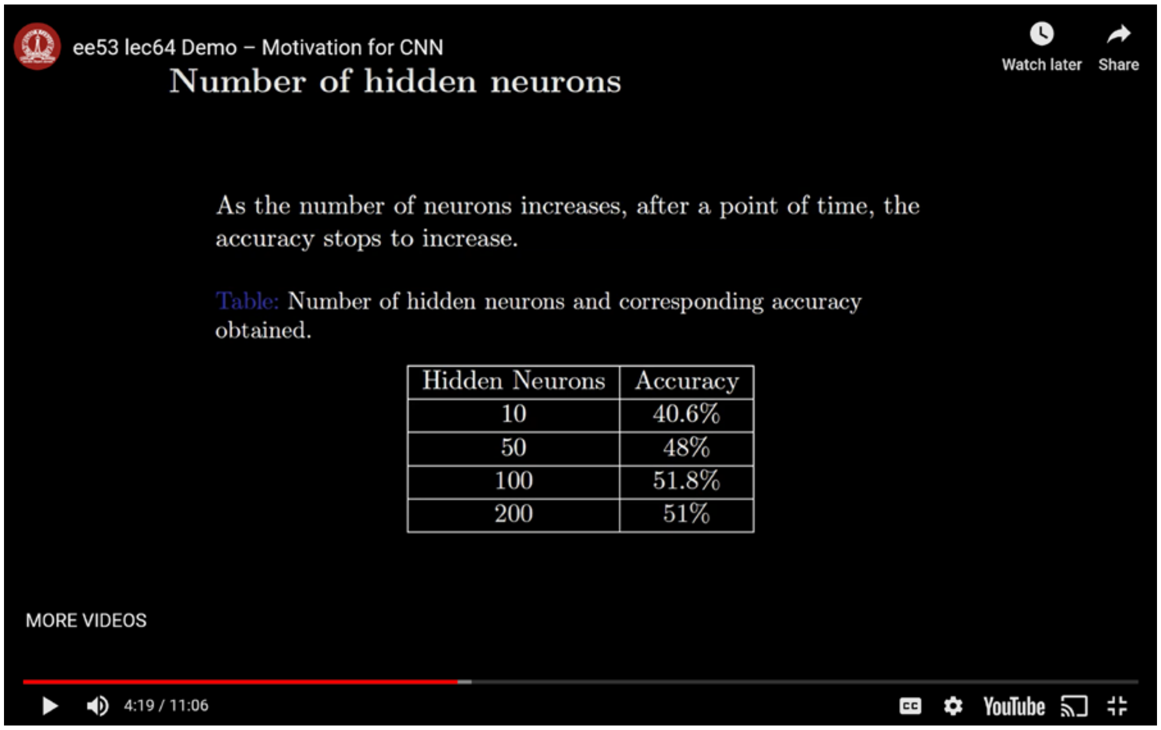
Consider a network with 100 hidden neurons. The image size is 32x32x3, which equals 3,072 pixels. Consequently, the number of parameters between the input layer and the hidden layer is 307,200, a substantial number, given that we only have 50,000 training samples. If the image size were increased to 960x70x3, 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)



Number of hidden neurons

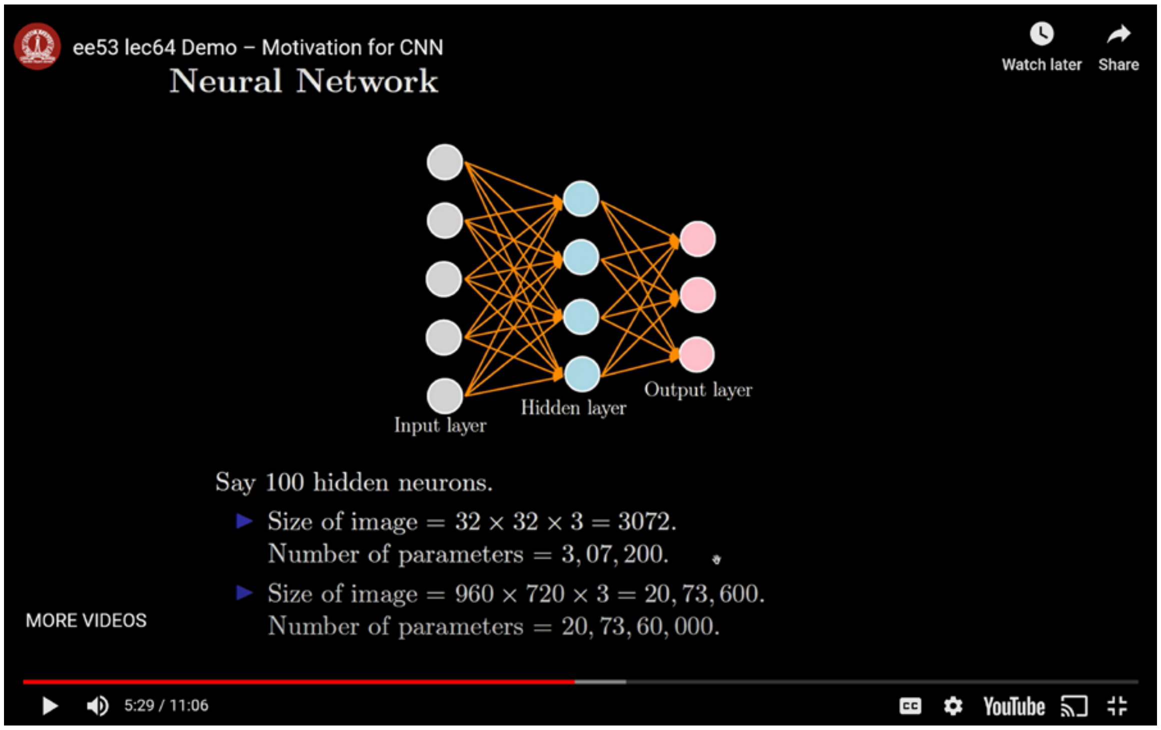
As the number of neurons increases, after a point of time, the accuracy stops to increase.

Table: Number of hidden neurons and corresponding accuracy obtained.

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

In convolution, given a 32x32x3 image, consider a filter of size 5x5x3. 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 32x32x3 image convolved with a 5x5x3 filter, the activation map size is 28x28x1. If we use 6 such filters, the output will be a 28x28x6 activation map.

(Refer Slide Time: 05:29)

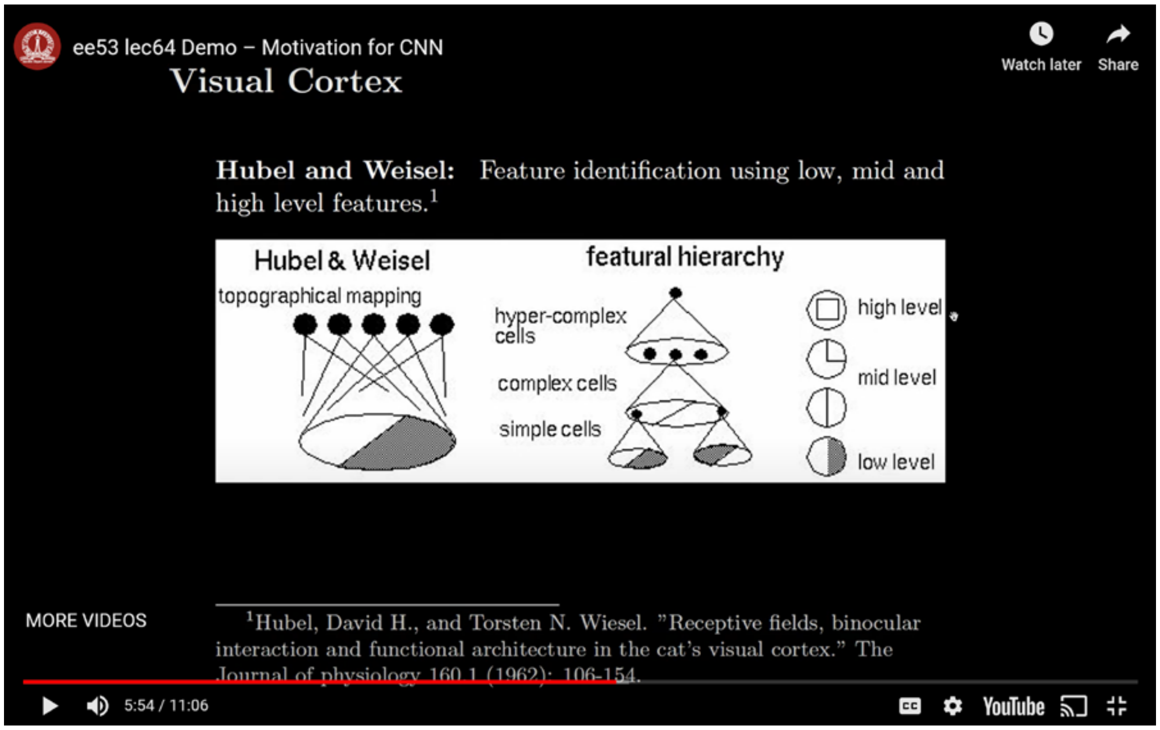


Neural Network

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 = $20,73,60,000$.

(Refer Slide Time: 05:54)



Visual Cortex

Hubel and Weisel: Feature identification using low, mid and high level features.[1]

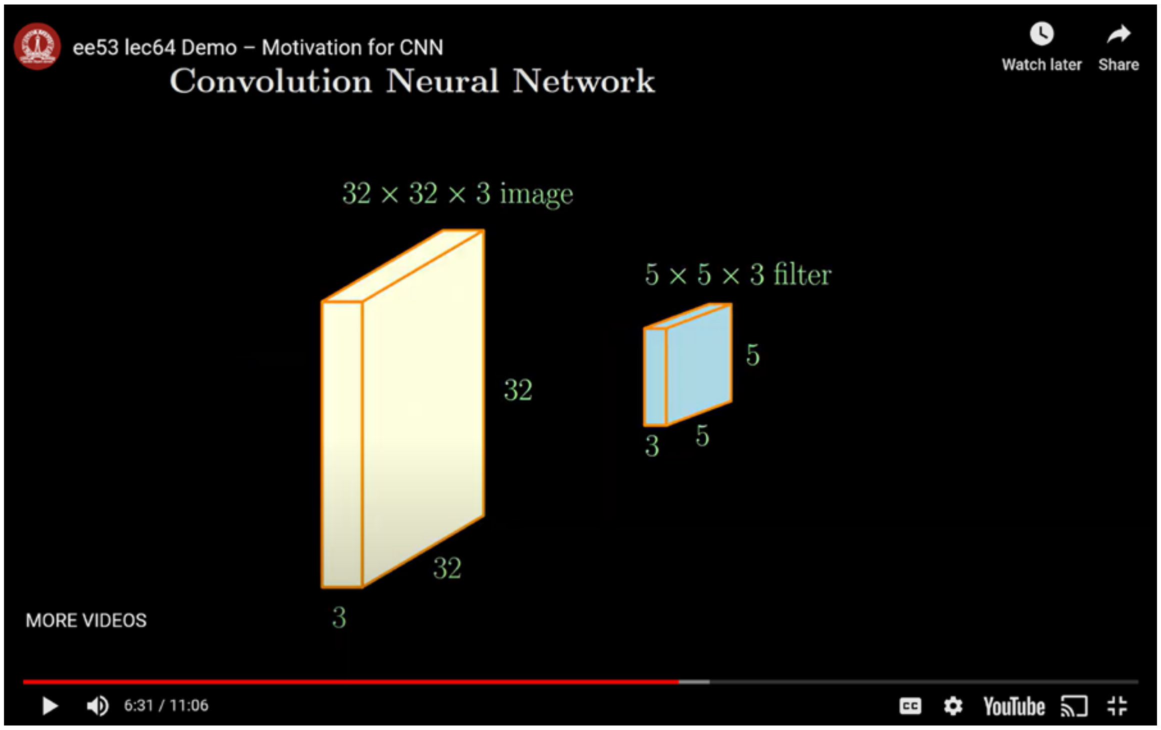
[1]Hubel, David H., and Torsten N. Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex." The Journal of physiology 160.1 (1962): 106-154.

This activation map becomes the new input with a size of 28x28x6. In a CNN architecture, multiple convolution layers are stacked sequentially for feature extraction. For instance, with six 5x5x3 filters, the output is 28x28x6. In the subsequent layer, if we use ten 5x5x6 filters, the output size becomes 24x24x10, 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 5x5x3 and 5x5x6 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 2x2 grid with a stride of 2. For each 2x2 grid, we identify and retain only the maximum value while discarding the remaining values.

(Refer Slide Time: 06:31)

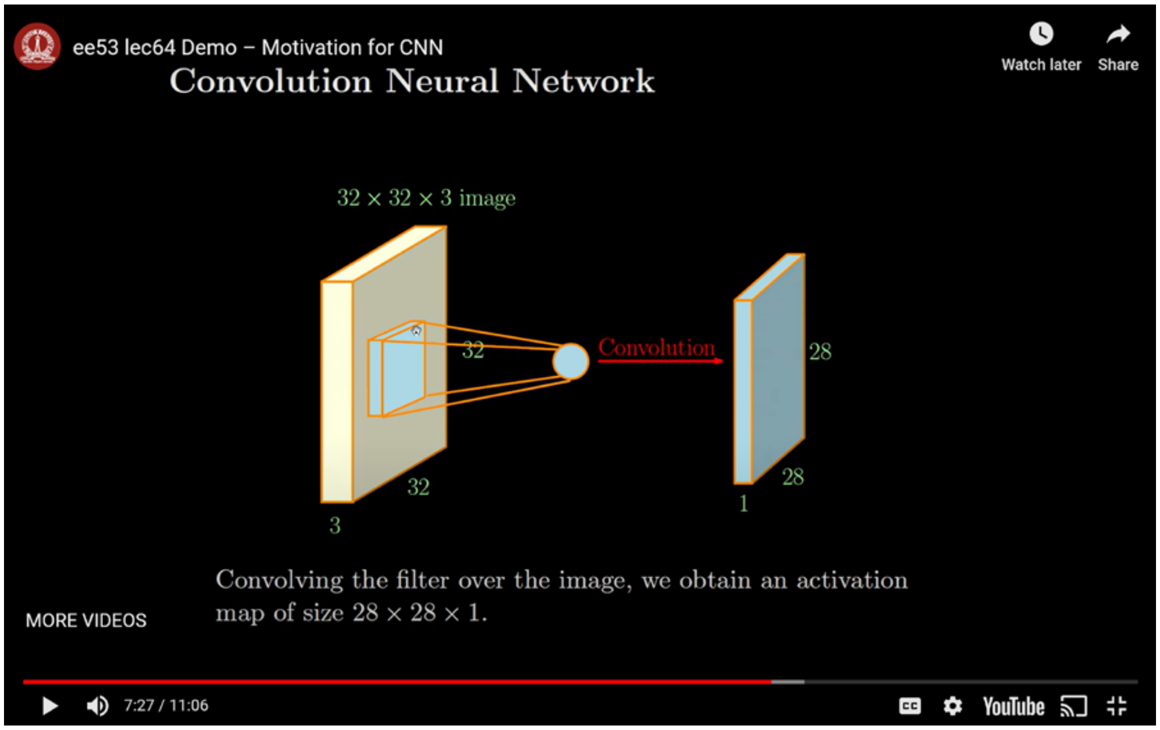


Convolution Neural Network

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)

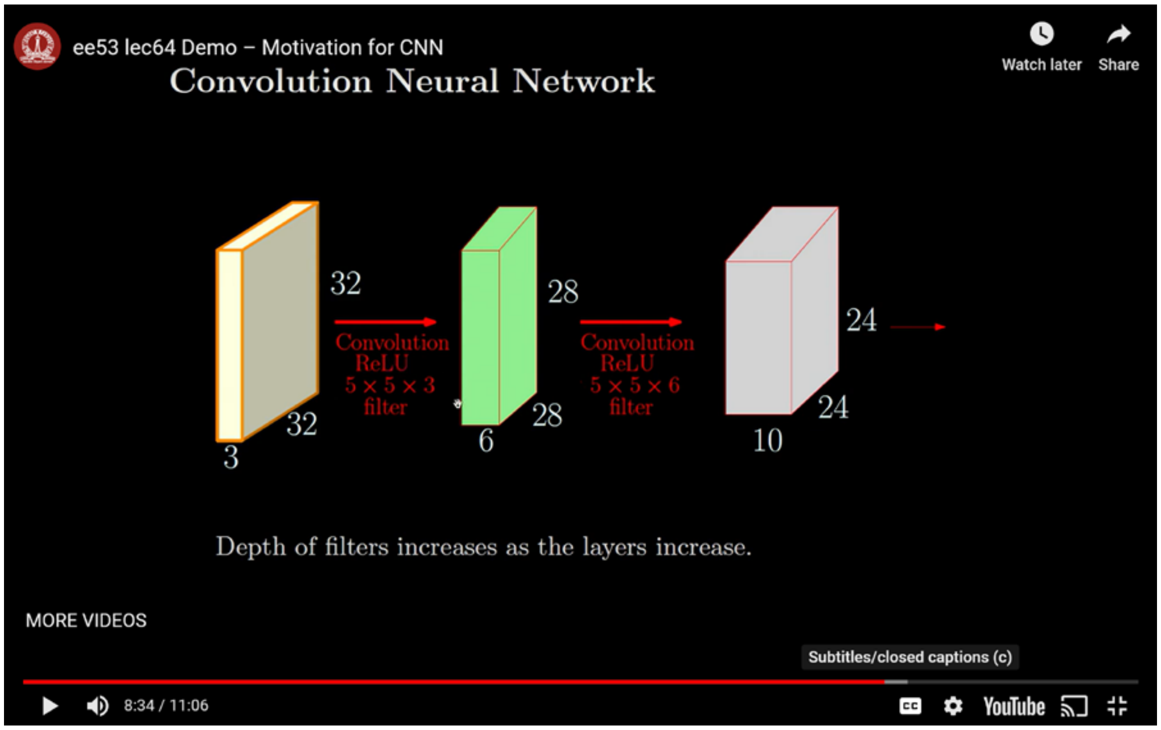

ee53 lec64 Demo – Motivation for CNN

Convolution Neural Network

Convolving the filter over the image, we obtain an activation map of size $28 \times 28 \times 1$.

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.

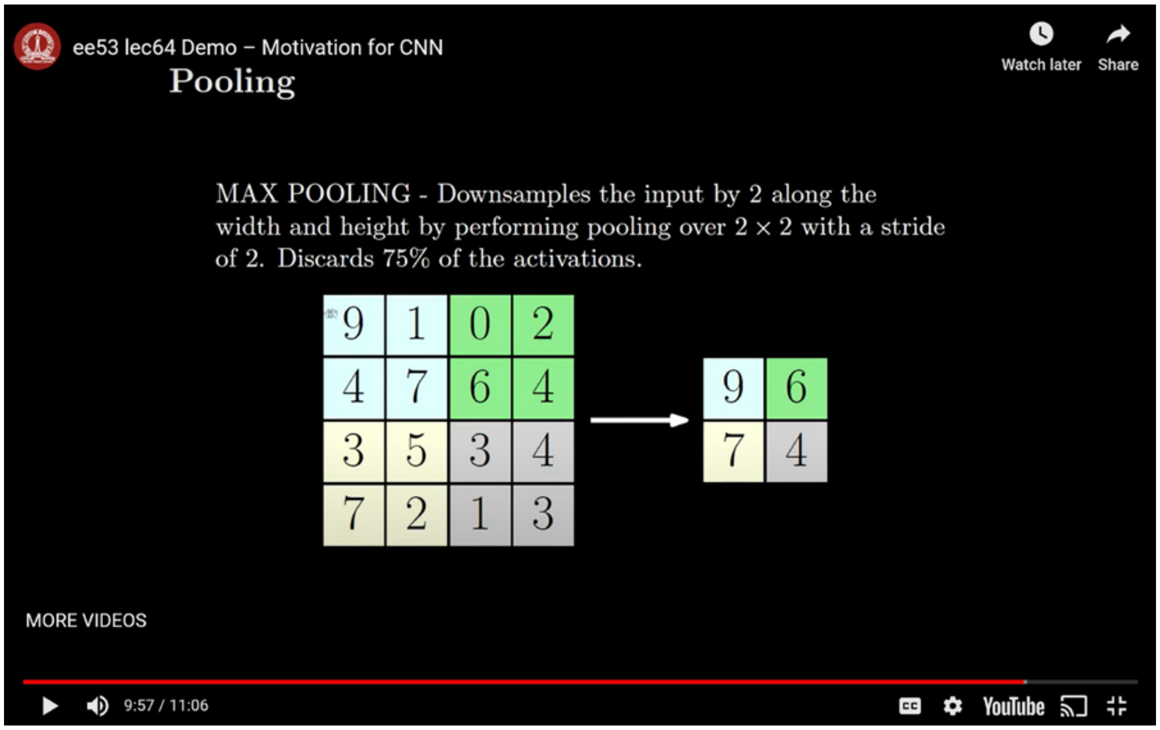
(Refer Slide Time: 08:34)



Convolution Neural Network

Depth of filters increases as the layers increase.

(Refer Slide Time: 09:57)

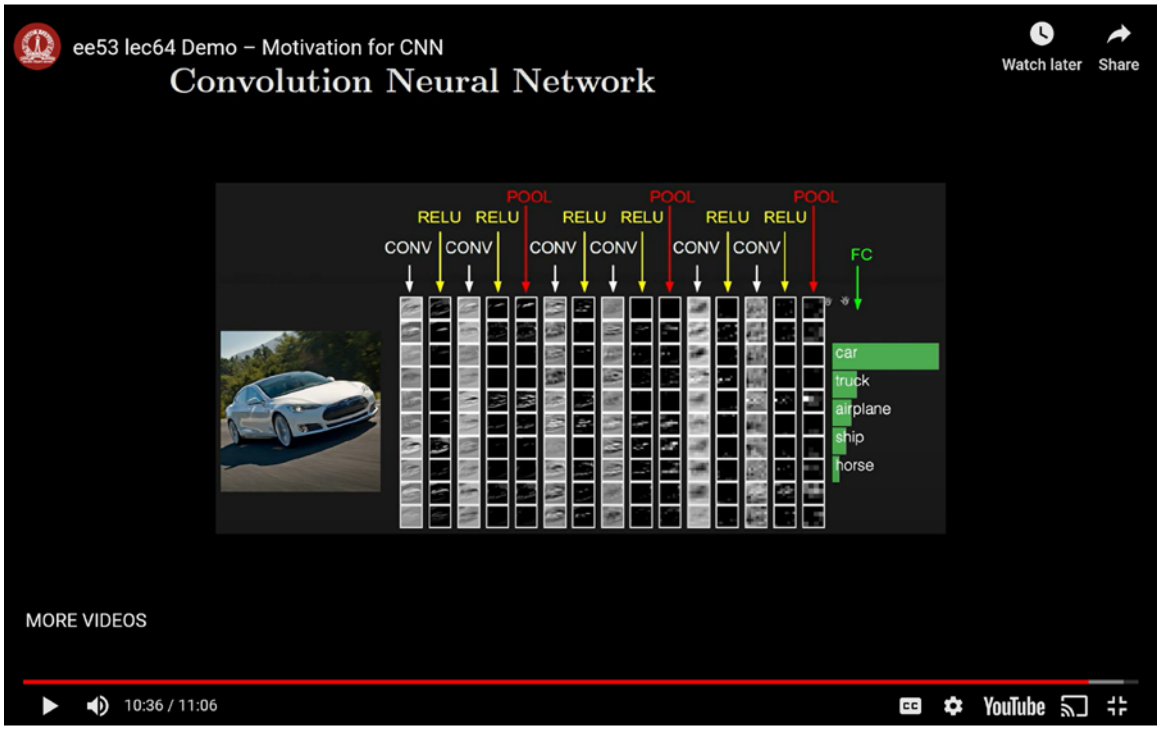


Pooling

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

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)



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.