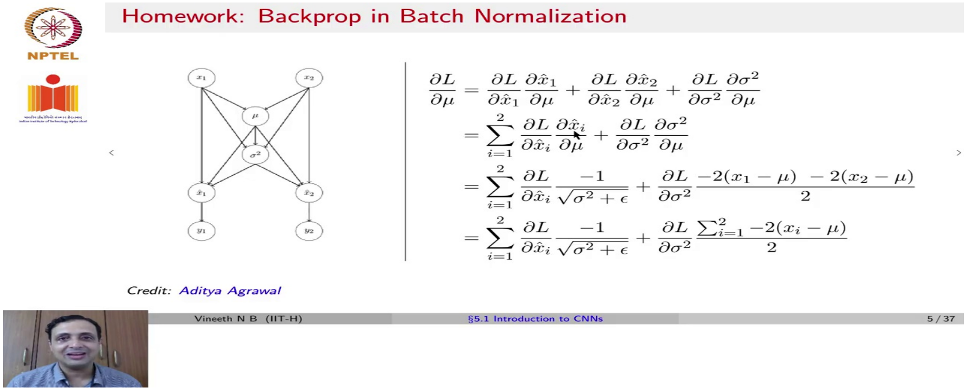


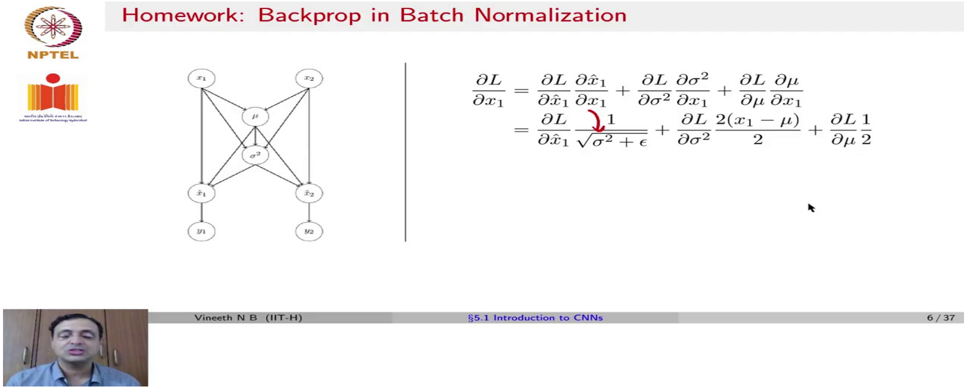
Homework: Backprop in Batch Normalization

$$
\begin{aligned}
\frac{\partial L}{\partial \mu} &= \frac{\partial L}{\partial \hat{x}_1}\frac{\partial \hat{x}_1}{\partial \mu} + \frac{\partial L}{\partial \hat{x}_2}\frac{\partial \hat{x}_2}{\partial \mu} + \frac{\partial L}{\partial \sigma^2}\frac{\partial \sigma^2}{\partial \mu} \\
&= \sum_{i=1}^{2} \frac{\partial L}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial \mu} + \frac{\partial L}{\partial \sigma^2}\frac{\partial \sigma^2}{\partial \mu} \\
&= \sum_{i=1}^{2} \frac{\partial L}{\partial \hat{x}_i}\frac{-1}{\sqrt{\sigma^2+\epsilon}} + \frac{\partial L}{\partial \sigma^2}\frac{-2(x_1-\mu)\ -2(x_2-\mu)}{2} \\
&= \sum_{i=1}^{2} \frac{\partial L}{\partial \hat{x}_i}\frac{-1}{\sqrt{\sigma^2+\epsilon}} + \frac{\partial L}{\partial \sigma^2}\frac{\sum_{i=1}^{2} -2(x_i-\mu)}{2}
\end{aligned}
$$

Credit: *Aditya Agrawal*

Vineeth N B (IIT-H) §5.1 Introduction to CNNs 5 / 37

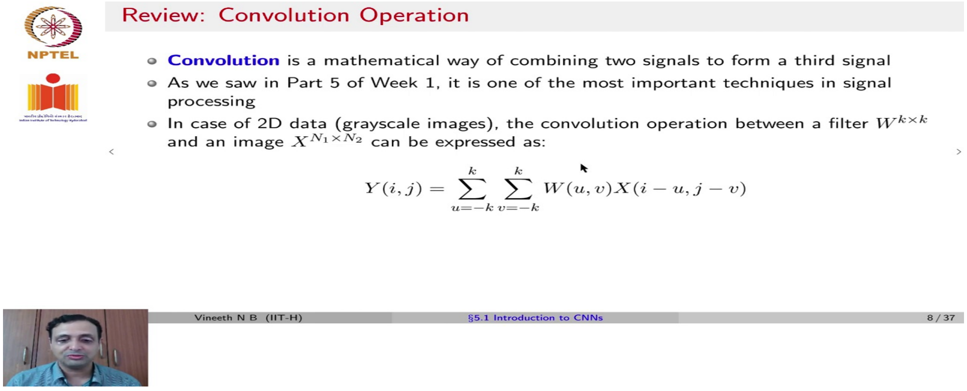
Note here, that in the last line of the algorithm, beta plus gamma times xi hat is the final output yi.So, you can assume that beta is almost an input into y with, so you have this here, gamma times xi hat plus beta times 1 is what is y, y1.So, which means doh L by doh beta is going to be given by doh L by doh y1 into doh y1 by doh beta.Doh L by doh x1 hat will be doh L by doh y1 into do y1 by doh x1 hat, doh y1 by doh x1 hat is given by gamma as we just saw, that is what we put in here.So, the gradient of doh xi hat by doh sigma just follows the gradient of this particular term which can be written by xi minus mu into minus half into sigma squared plus epsilon which would have been the denominator into -3/2.Similarly, we have doh L by doh mu which is doh L by doh x1 hat into doh x1 hat by doh mu plus doh L by doh x2 hat into doh x2 hat by doh mu and also doh L by doh sigma square into doh sigma square by doh mu.Doh xi hat by doh mu is given by this gradient here; this once again follows directly from the definition of xi hat to be xi minus mu by root sigma square plus epsilon.



Homework: Backprop in Batch Normalization

$$
\begin{aligned}
\frac{\partial L}{\partial x_1} &= \frac{\partial L}{\partial \hat{x}_1}\frac{\partial \hat{x}_1}{\partial x_1} + \frac{\partial L}{\partial \sigma^2}\frac{\partial \sigma^2}{\partial x_1} + \frac{\partial L}{\partial \mu}\frac{\partial \mu}{\partial x_1} \\
&= \frac{\partial L}{\partial \hat{x}_1}\frac{1}{\sqrt{\sigma^2+\epsilon}} + \frac{\partial L}{\partial \sigma^2}\frac{2(x_1-\mu)}{2} + \frac{\partial L}{\partial \mu}\frac{1}{2}
\end{aligned}
$$

Vineeth N B (IIT-H) §5.1 Introduction to CNNs 6 / 37

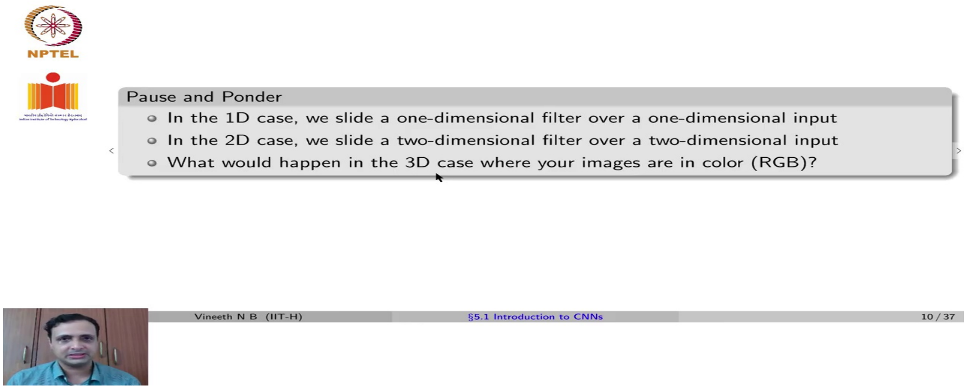


Review: Convolution Operation

- **Convolution** is a mathematical way of combining two signals to form a third signal
- As we saw in Part 5 of Week 1, it is one of the most important techniques in signal processing
- In case of 2D data (grayscale images), the convolution operation between a filter $W^{k \times k}$ and an image $X^{N_1 \times N_2}$ can be expressed as:

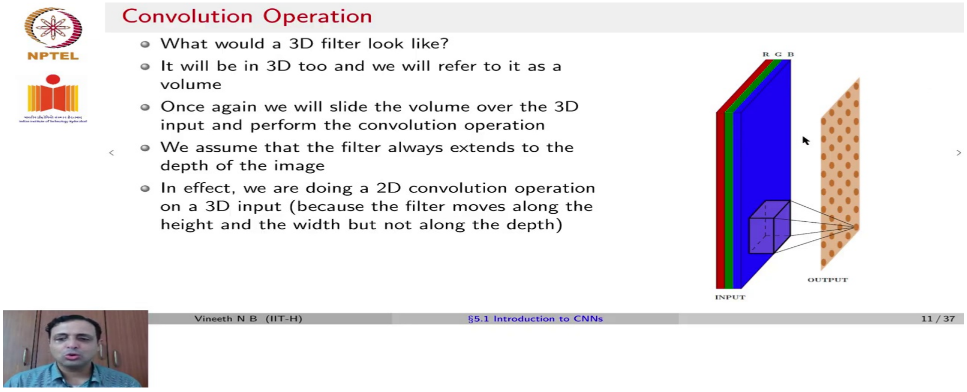
$$Y(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} W(u,v)X(i-u, j-v)$$

Vineeth N B (IIT-H) §5.1 Introduction to CNNs 8 / 37



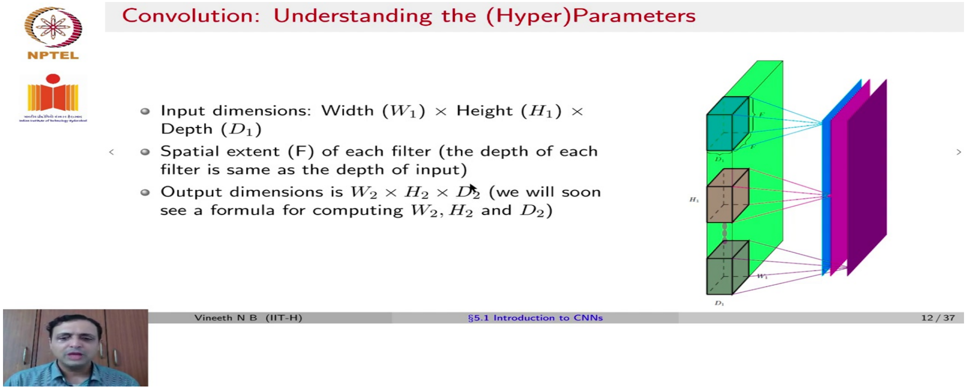
Pause and Ponder

- In the 1D case, we slide a one-dimensional filter over a one-dimensional input
- In the 2D case, we slide a two-dimensional filter over a two-dimensional input
- What would happen in the 3D case where your images are in color (RGB)?



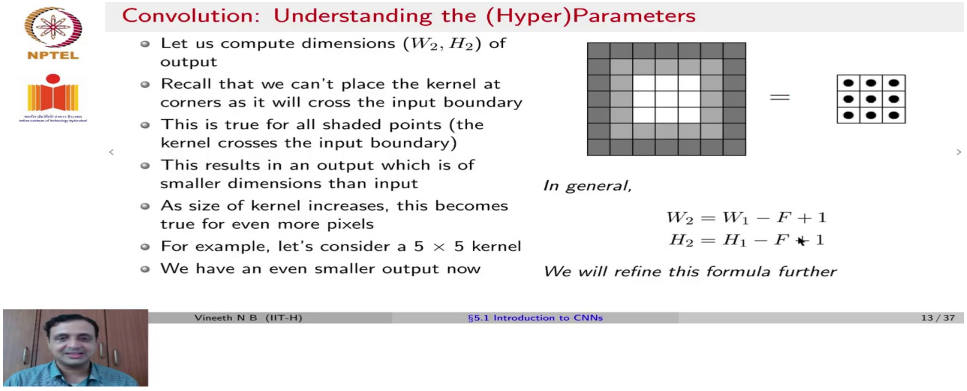
Convolution Operation

- What would a 3D filter look like?
- It will be in 3D too and we will refer to it as a volume
- Once again we will slide the volume over the 3D input and perform the convolution operation
- We assume that the filter always extends to the depth of the image
- In effect, we are doing a 2D convolution operation on a 3D input (because the filter moves along the height and the width but not along the depth)



Convolution: Understanding the (Hyper)Parameters

- Input dimensions: Width (W_1) × Height (H_1) × Depth (D_1)
- Spatial extent (F) of each filter (the depth of each filter is same as the depth of input)
- Output dimensions is $W_2 \times H_2 \times D_2$ (we will soon see a formula for computing W_2, H_2 and D_2)



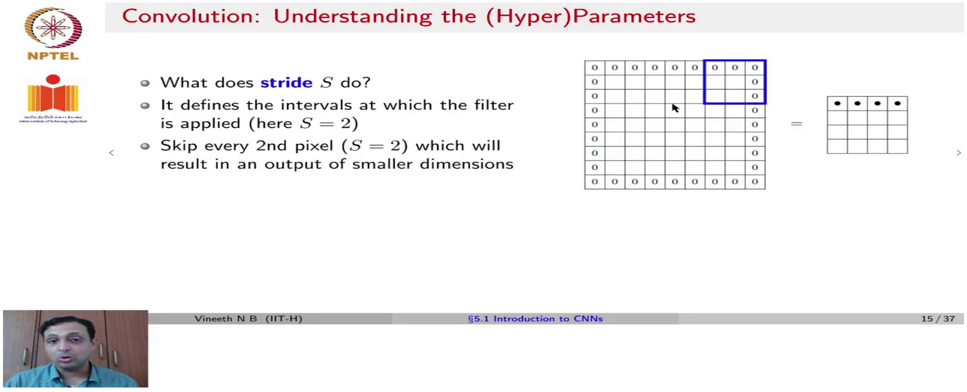
Convolution: Understanding the (Hyper)Parameters

- Let us compute dimensions (W_2, H_2) of output
- Recall that we can't place the kernel at corners as it will cross the input boundary
- This is true for all shaded points (the kernel crosses the input boundary)
- This results in an output which is of smaller dimensions than input
- As size of kernel increases, this becomes true for even more pixels
- For example, let's consider a 5 × 5 kernel
- We have an even smaller output now

In general,

$$W_2 = W_1 - F + 1$$
$$H_2 = H_1 - F + 1$$

We will refine this formula further



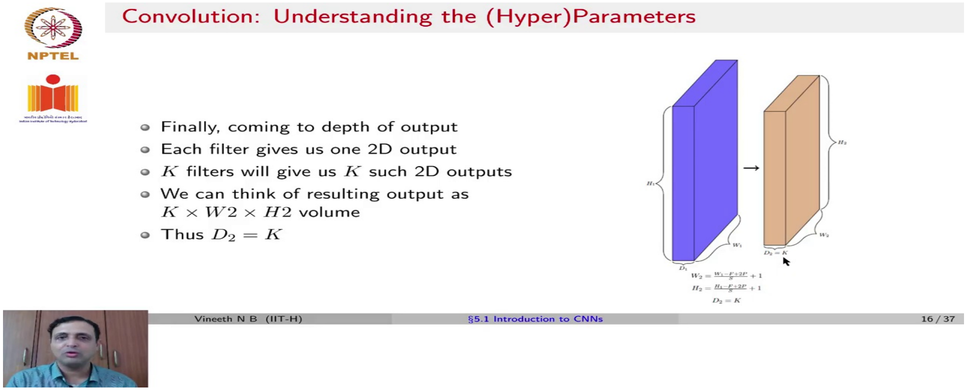
Convolution: Understanding the (Hyper)Parameters

- What does **stride** S do?
- It defines the intervals at which the filter is applied (here $S = 2$)
- Skip every 2nd pixel ($S = 2$) which will result in an output of smaller dimensions

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

=

●	●	●	●



Convolution: Understanding the (Hyper)Parameters

- Finally, coming to depth of output
- Each filter gives us one 2D output
- K filters will give us K such 2D outputs
- We can think of resulting output as $K \times W2 \times H2$ volume
- Thus $D_2 = K$

$$W_2 = \frac{W_1 - F + 2P}{S} + 1$$
$$H_2 = \frac{H_1 - F + 2P}{S} + 1$$
$$D_2 = K$$