CS 331
Computer Vision

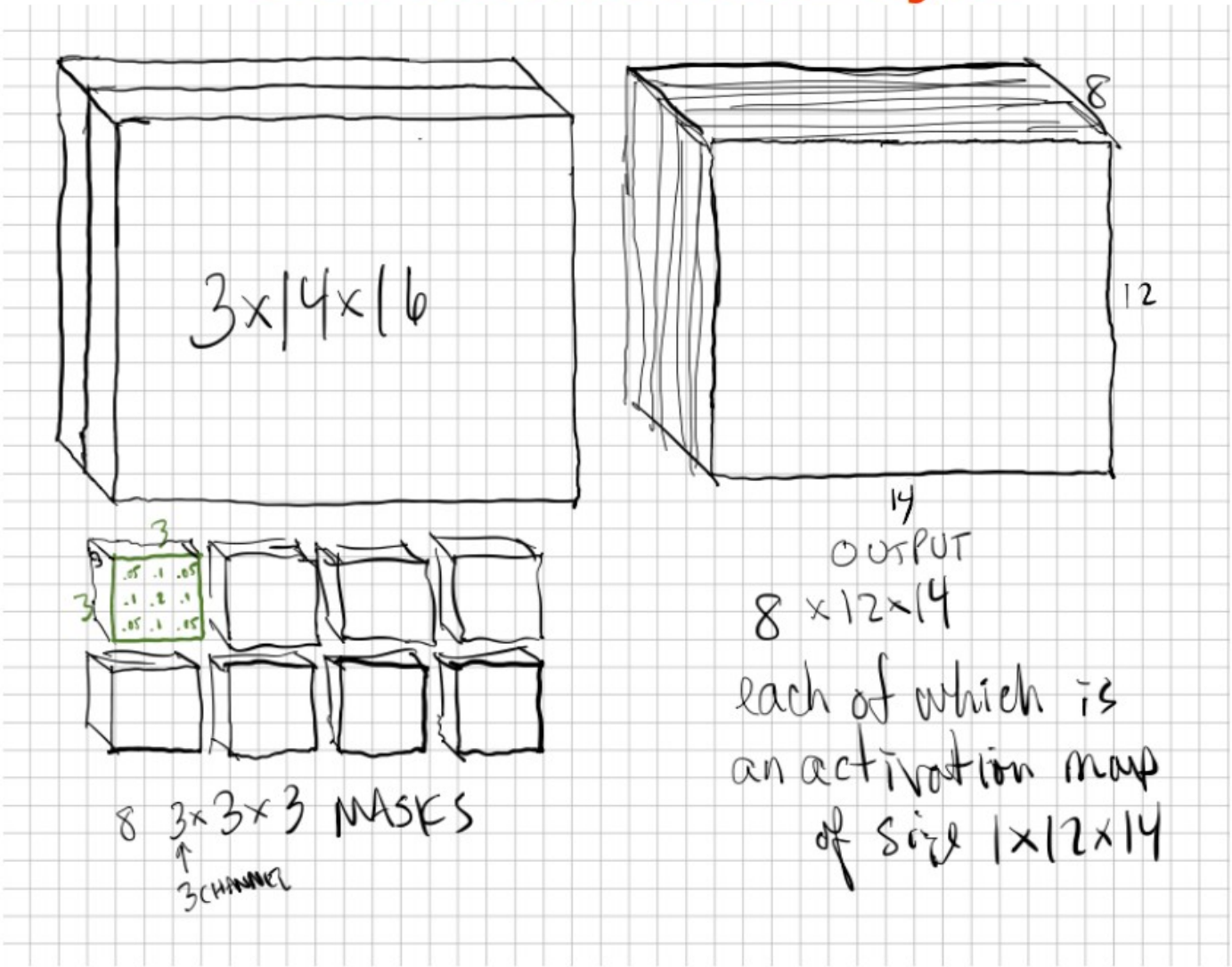
11 / 12 / 2026

Instructor: Michael Eckmann

Topics

- Questions / Comments?
- More Convolutional Neural Networks
  - How to determine the number of parameters and the sizes of outputs etc.

Convolutional Layer



Convolutional Layer

- In the example on last slide:
- Each of the 8 masks has 27 elements.
- Each 3x3x3 mask will be elementwise multiplied by a 3x3x3 chunk of the input image.
- One output value would be computed by doing 27 multiplies and 26 adds.

Another note: what I am referring to as a mask is often called a filter or a kernel in the literature. So I use these terms interchangeably.

Tensor

- Think of a tensor as a multidimensional array.
- A scalar is a 0-dimensional tensor
- A vector is a 1-dimensional tensor
- A matrix is a 2-dimensional tensor
- But for higher dimensions we usually call them tensors and the number of dimensions can be specified.

Convolutional Layer

- If there were 1 image with Cin channels, inRows rows and inCols columns the input is a 3 dimensional tensor of shape (Cin, inRows, inCols).
- Each mask will be of shape (Cin, maskRows, maskCols) and there could be Cout of them which could be stored in a 4-d tensor of shape (Cout, Cin, maskRows, maskCols).
- The output of the convolutional layer would produce a 3-dimensional tensor of shape (Cout, outRows, outCols).
- Last picture had: (Cin, inRows, inCols) = (3,14,16)
- (Cout, Cin, maskRows, maskCols) = (8,3,3,3)
- (Cout, outRows, outCols) = (8,12,14)
- These are specified channels-first style

Convolutional Layer

- More often we use a batch of N images as input.
- The input would now be a 4-d tensor of shape:

  (N, inRows, inCols, Cin).
- The masks would be stored in a 4-d tensor of shape

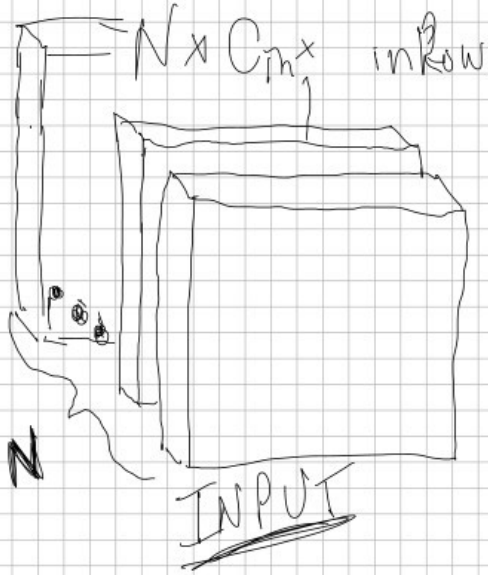
  (Cout, maskRows, maskCols, Cin).

  Cout = # of masks
- The output of the convolutional layer would produce a 4-d tensor of shape

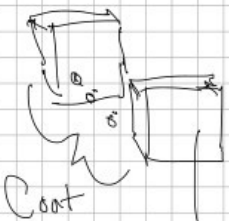
  (N, outRows, outCols, Cout).

Here I stated them in channels-last style.

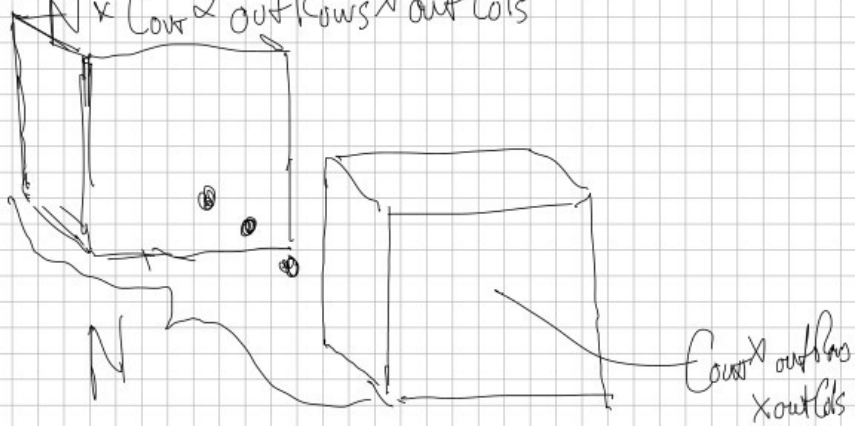
INPUT A BATCH OF N IMAGES ALL OF SAME # CHANNELS, # ROWS & # COLS.



Cout # of MASKS
each of size Cin x Mr x Mc



OUTPUT:
N x Cout x outRows x outCols



Convolutional Layer

- EXAMPLE: Suppose batch of 16 images as input, each 3 channel and 32x32.
- The input would now be a 4-d tensor of shape:

  (16, 32, 32, 3).
- The masks say are each of size 5x5 with 3 channels. And we have, say, 8 masks. 4-d tensor of shape:

  (8, 5, 5, 3).
- The output of the convolutional layer would produce a 4-d tensor of shape

  (16, 28, 28, 8).

Convolutional Layer

- Each mask will have a bias unit associated with it. So for the example of last slide, we will have 8 bias units.
- The # of weights (the number of learnable parameters) is Cout * ((Cin * maskRows * maskCols) + 1)
- Last slide: 8 * ((3*5*5) + 1) = 8*76 = 608
- Note the difference in number of weights vs. a Fully Connected layer.

Convolutional Layer

- We add an activation function between layers for non-linearity purposes. So it is common to send the weights to a ReLU before the next layer.

AlexNet

- AlexNet, published in 2012, is the first convolutional neural network to win the ImageNet challenge. This sparked the use of CNNs moving forward in computer vision.
- Citation: “ImageNet classification with deep convolutional neural networks”, Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, NIPS 2012.

AlexNet

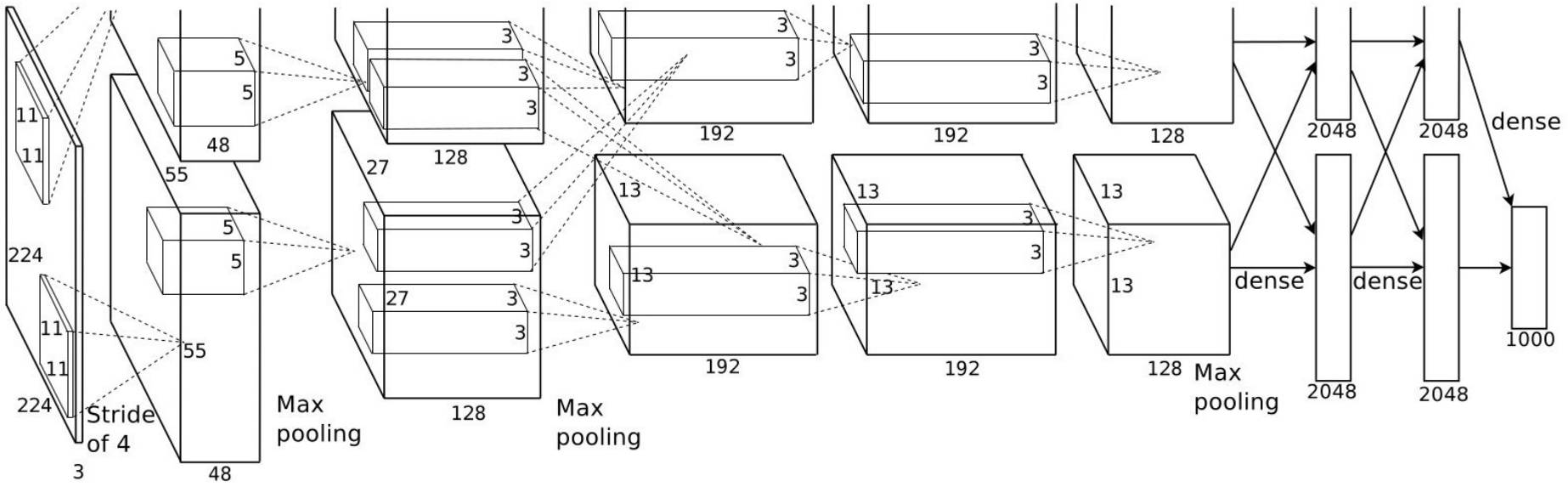


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

Learned Masks

- Let’s talk about the meaning of a learned mask (aka kernel). Think back to when we first learned convolution --- we convolved an image with a horizontal edge mask.
- This mask responded highly where the input image resembled a horizontal edge.
- By looking at the learned masks visually we can get some indication of what they are learning

AlexNet

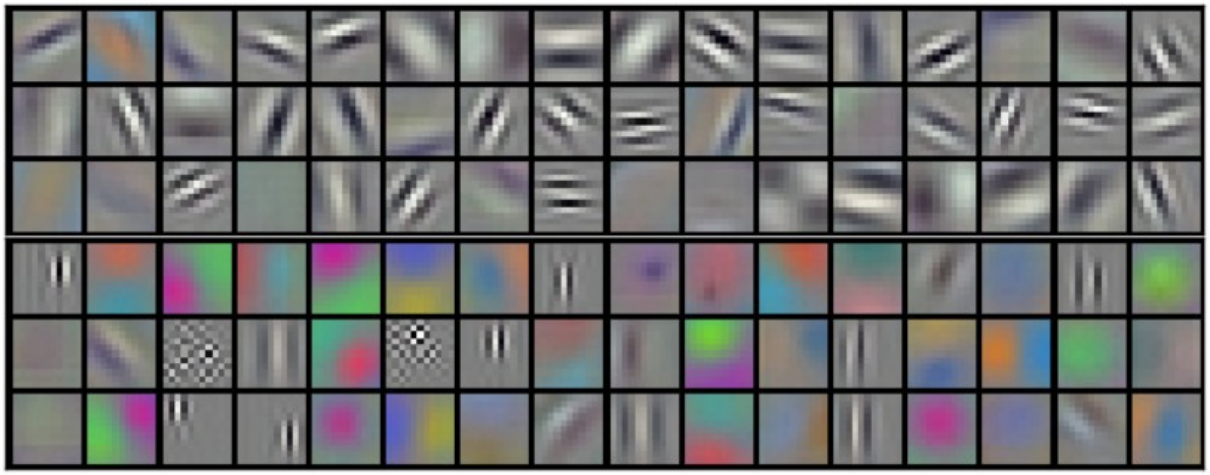


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.