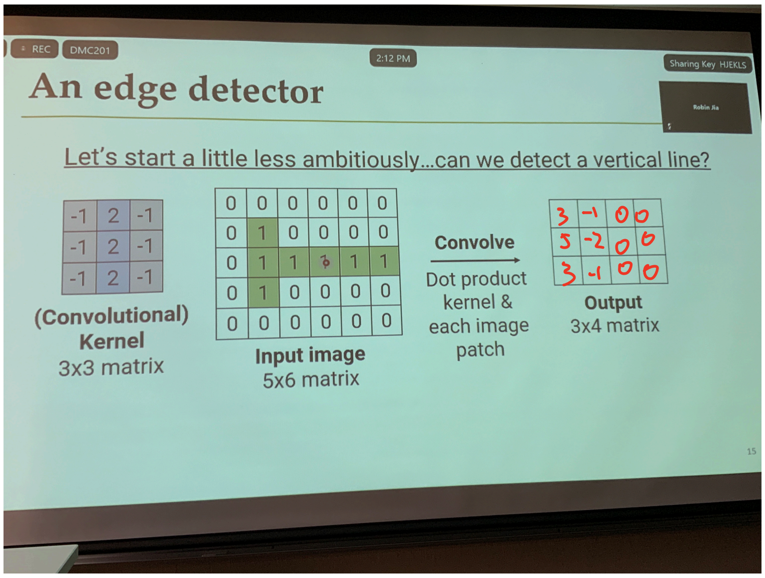
Process images by learning features hierarchically
Start with most basic features on smallest patches
Based on those, identify more complex features

Outline

- Extracting features with convolutions
- CNN
- Computer vision tasks



REC DMC201 2:12 PM Sharing Key HJEKLS

Robin Jia

An edge detector

Let's start a little less ambitiously...can we detect a vertical line?

-1	2	-1
-1	2	-1
-1	2	-1

(Convolutional) Kernel
3x3 matrix

0	0	0	0	0	0
0	1	0	0	0	0
0	1	1	1	1	1
0	1	0	0	0	0
0	0	0	0	0	0

Input image
5x6 matrix

Convolve →
Dot product kernel & each image patch

3	-1	0	0
5	-2	0	0
3	-1	0	0

Output
3x4 matrix

15

Convolution takes in two matrices

- kernel k by k
- Input w by h
- Output w-k-1 by h-k-1 matrix
- Convolutional layer
    - Kernel is weight/parameter
    - Use convolution to extract features
- A linear operation

Motivation #1: local receptive fields

- each neuron only looks at a small patch of input
- why? Local texture/shapes are useful
- Understand from local -> global patterns

Motivation #2: weight sharing

- in each local receptive field, the same types of features are useful
    - Basic: detecting edges
    - More advanced: detecting moos
- Share the same kernel
- Convolutions encode translation equivalence

CNN vs MLP

- CNN fewer parameters => need less data to learn useful features
- MLP have to learn to detect the same feature over and over again at different locations

Multiple input channels

- color image has 3 “channels” for red/green/blue
- Inputs is actually 3 x w x h
- Kernel has size Cin x k x k

Multiple output channels

- can have multiple kernels, each to detect a different thing
- One for vertical lines, one for horizontal lines, etc.
- Total size of kernel tensor is Cout x Cin x k x k

Stride and padding

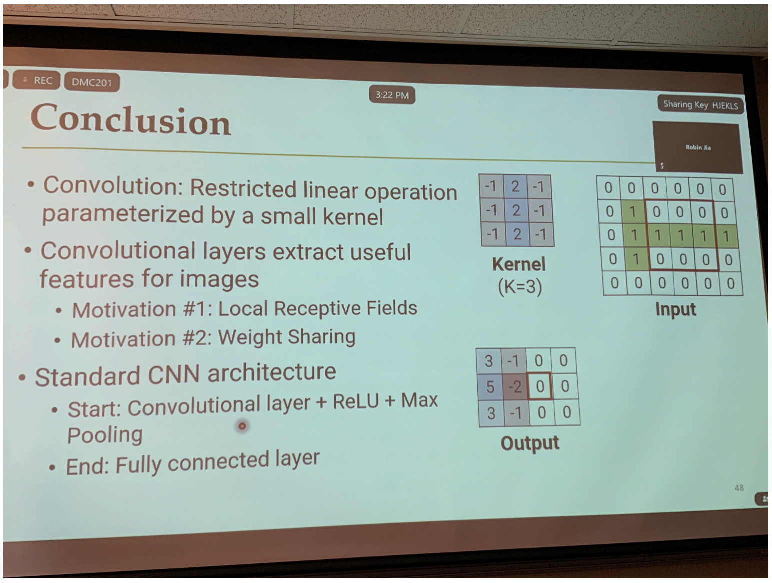
- Stride: as you slide across image, how big of a step do you take
    - Default: stride = 1 pixel
    - Can choose larger stride to reduce dimensionality
- padding: pad the edges of images with 0’s
    - For k=3 and no padding, width/height shrink by 2 each time
    - Adding width-1 padding on each side prevent this
    - For k=5, pad by 2
    - Default: no padding

Convolutional layers

- Convolution + ReLU
- Pooling
    - Look for larger features
    - Reduce resolution of input by a factor of P (often P=2)
        - Average pool: average in each 2x2 patch
        - Max pool: max in each 2x2 patch
- Flatten
- Fully connected
- Softmax

Computer vision

- object detection
- Semantic segmentation



Conclusion

- Convolution: Restricted linear operation parameterized by a small kernel
- Convolutional layers extract useful features for images
    - Motivation #1: Local Receptive Fields
    - Motivation #2: Weight Sharing
- Standard CNN architecture
    - Start: Convolutional layer + ReLU + Max Pooling
    - End: Fully connected layer

-1	2	-1
-1	2	-1
-1	2	-1

Kernel
(K=3)

0	0	0	0	0	0
0	1	0	0	0	0
0	1	1	1	1	1
0	1	0	0	0	0
0	0	0	0	0	0

Input

3	-1	0	0
5	-2	0	0
3	-1	0	0

Output

48