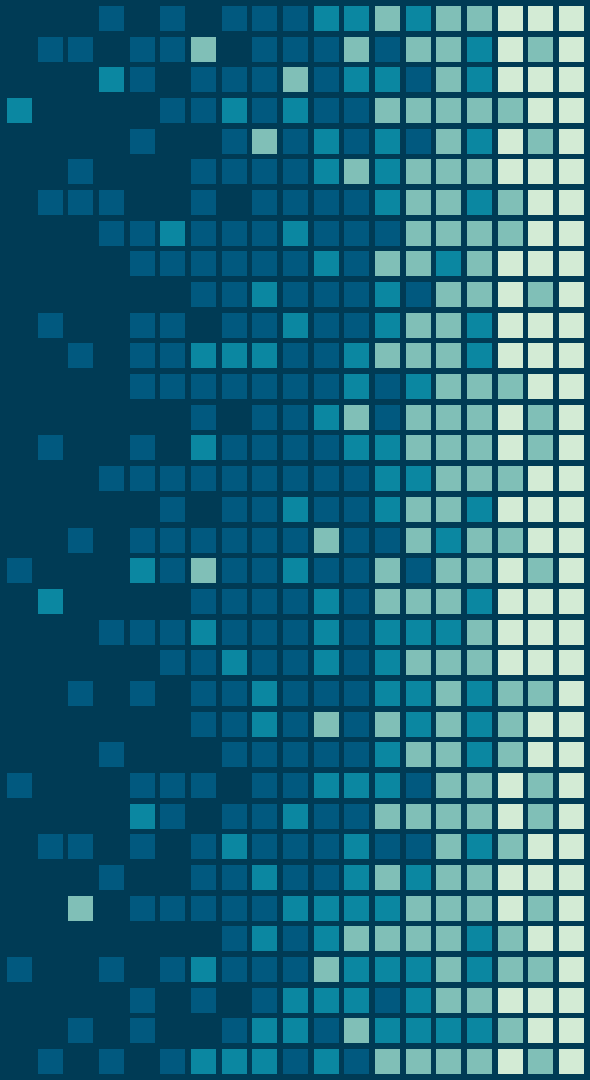
ACM AI

Intro to Machine Learning

Advanced Track Workshop #3: Intro to Convolutional Neural Networks



ACM AI

SIGN IN!

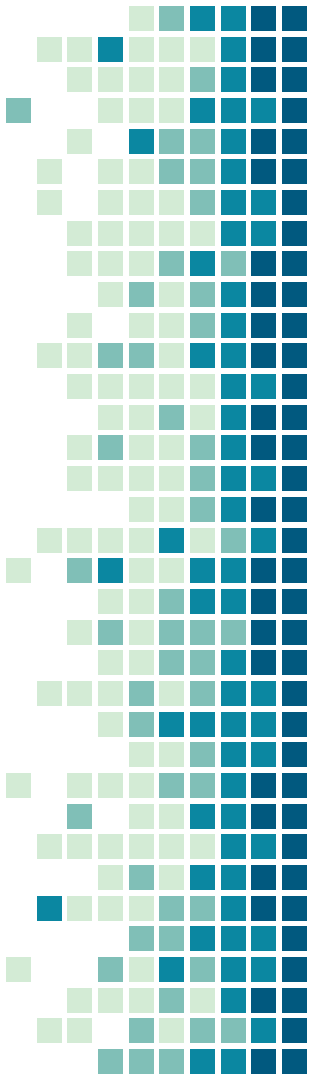
ACM AI

Resources

Slides: https://tinyurl.com/advtrack-fall19-w5

Colab: colab.research.google.com

Membership Portal Code: atrack3



Today's Contents

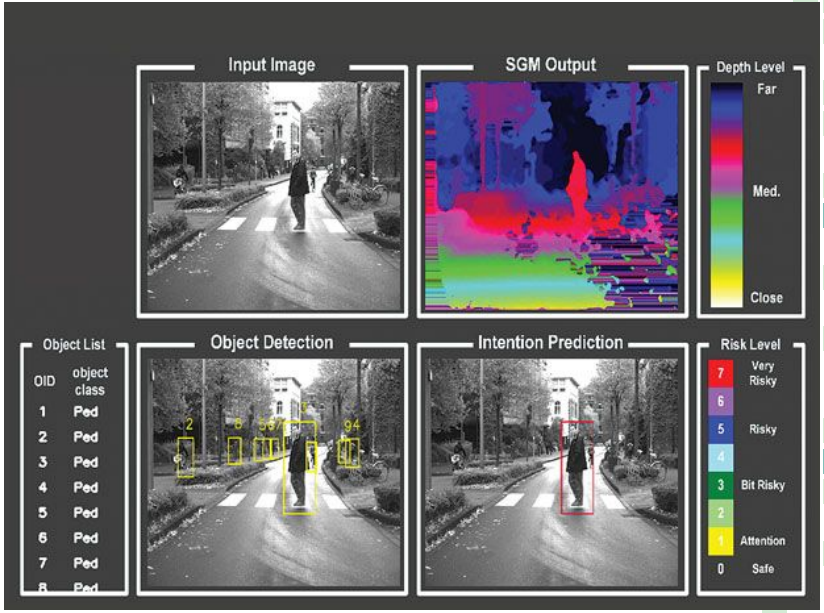
- Applications of CNNs
- Theory and Intuition for CNNs
  - brief review
  - convolutional layer
  - other layers

ACM AI

2. Applications of Conv Nets

Conv Nets Applications

- Used in self driving cars to make predictions on the optimal course of action by companies developing them.



Facebook tagging

- Facebook uses Conv Nets multiple times to be able to accurately tag people in photos by picking out individual faces and classifying them.

3. Conv Nets - Context

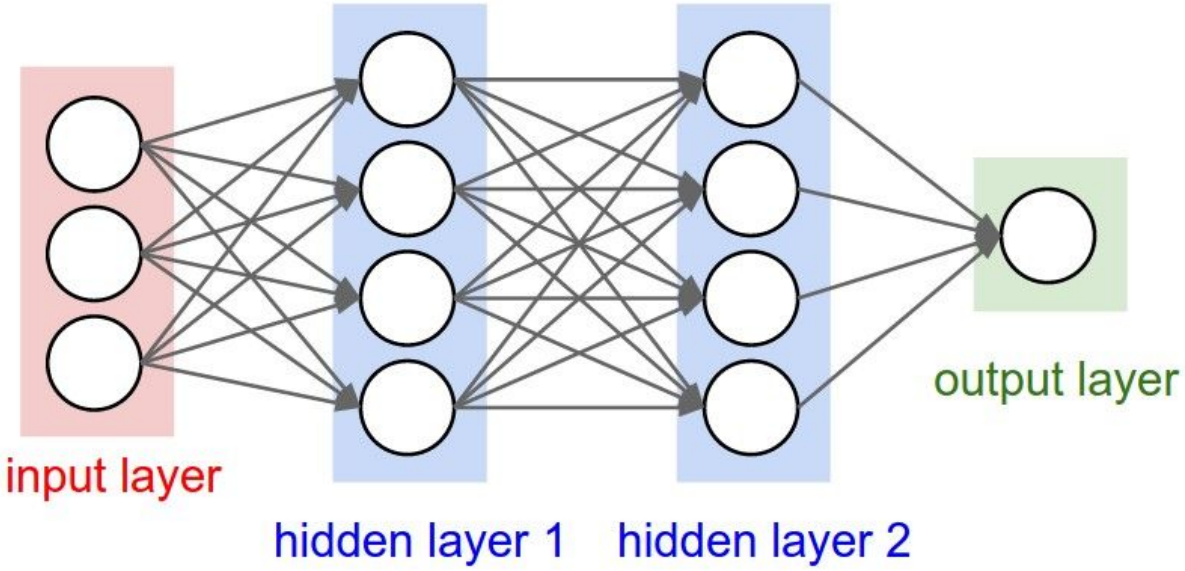
What is a CNN?

- A type of **neural network** used for computer vision
- The computer performs classification by looking for low level features, and using **convolutional layers** to build up to more complex, higher level features

Recap on Neural Nets

- Let's think about working with a single image first, without worrying about batch sizes.
- Regular NNs work primarily with "vectors" of input values (think 1D array containing pixel values), and "matrices" of weights (think 2D array).
- The output of any layer is a 1D vector which is then passed on to the next layer.

Recap on Regular NNs



Structure of CNNs

- CNNs work with ***volumes***.
- An image has a width and height, but also a 3rd dimension - the RGB values for each pixel
- The input is therefore 3D “matrix” of dimensions

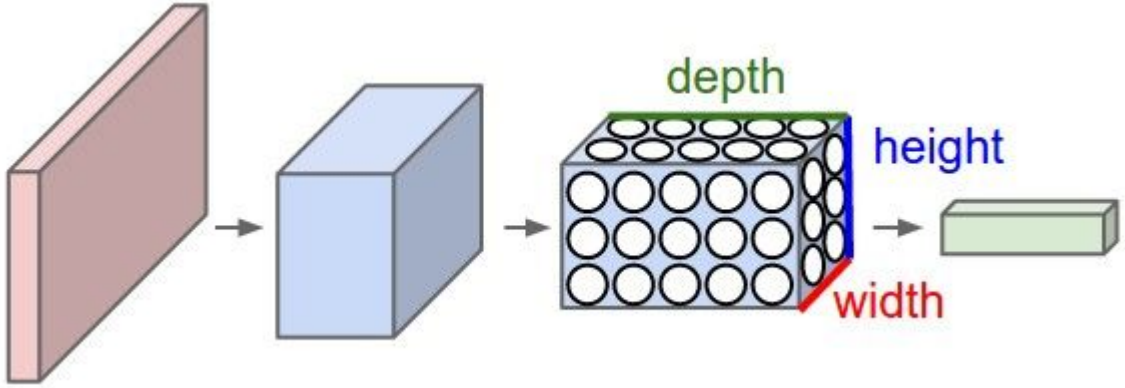
  Height x Width x 3 (usually)

- Here, the depth is 3 because each pixel has 3 “channels”, one for each of Red, Green and Blue.

Structure of CNNs

Like regular NNs, CNNs are made of **multiple layers**

Each layer takes some input volume and produces an output volume



Structure of CNNs: Layers

- Convolutional layer: creates a feature map
- ReLU: provides nonlinearity
- Pooling: downsamples
- Final layer: fully connected layer
  - Outputs an N-dimensional vector (N being the number of classes)

Formalizing different kinds of layers

- Though each layer takes a volume and gives a volume, the actual computation it performs depends on what kind of layer it is
- Common examples: **Convolutional**, **Pooling**, **ReLu** (or non-linear) and **Fully Connected**

“

Questions?

4. Convolutional Layer

Convolutional Layer – Filters

A filter is a 3D volume too. It is the volume of ***weights*** that acts on an input volume

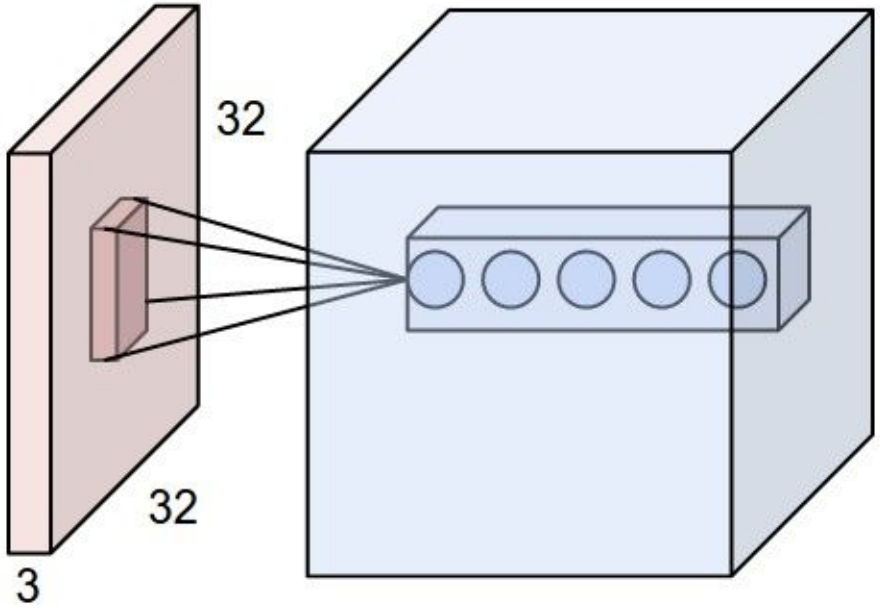
This “window” has the same depth as the input volume, but its height and width can vary (the area of a filter is usually much less than the spatial area of the input volume)

So if an input volume has the dimensions 28x28x10, the filter could have dimensions 5x5x10 or 7x7x10 and so on

Convolutional Layer – Filters

For a grayscale MNIST image input, which is 28x28, what would be a possible filter dimension to be used in our first convolutional layer?

Filter – Visualization



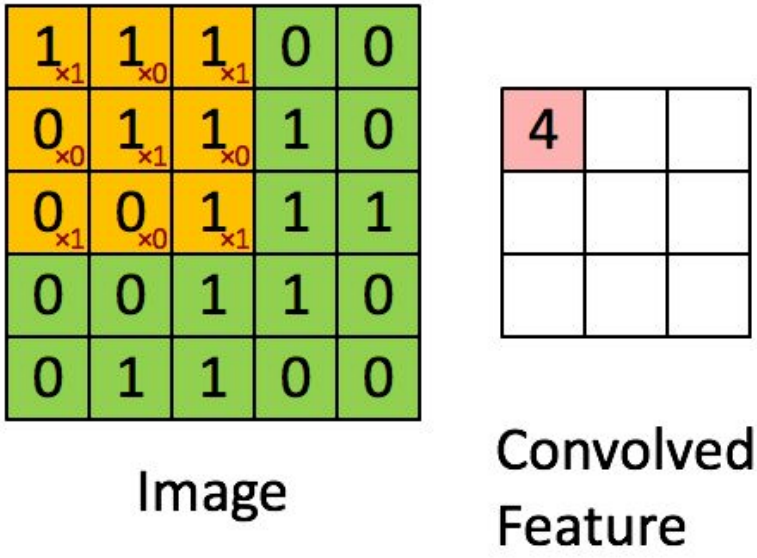
Filters – What do they do?

A filter is a volume of weights that moves across (convolves) the input volume, performing element-wise multiplications (dot products) at every place it “stops”

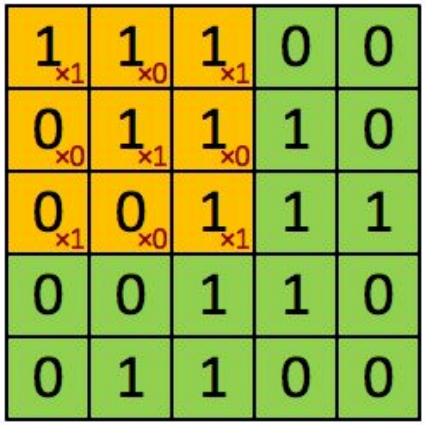
A filter moves with a **stride**. A stride of *n* means the filter moves by *n* positions before stopping to carry out a dot product.

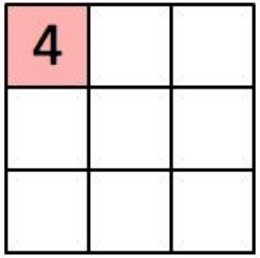
Every dot product results in a scalar. These numerous scalars make up the output volume

Filters – Visualization (ignoring depth)



What is the stride of this filter in this example?





Filters – What do they do?

A filter “scans” the input volume, resulting in a **2D** output.

A single filter acts on a volume and outputs an “area” - Why?

How then do we get an output **volume**?

Filters – What do they do?

A filter “scans” the input volume, resulting in a **2D** output.

A single filter acts on a volume and outputs an “area” - Why?

How then do we get an output **volume**?

We use multiple filters for a single input volume.

Convolutional Layers

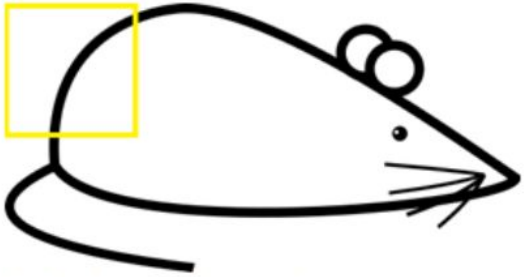
A Convolutional layer consists of multiple filters of the same shape acting on an input volume to produce an output volume.

Each filter is a set of weights that “looks for features” in the input volume, producing a higher value (greater activation) if that feature is found in a particular region.

A bias vector of size K (number of filters) is also used

Visualization

Original image



Visualization of the filter on the image

Visualization of the receptive field

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Pixel representation of the receptive field

*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

Multiplication and Summation = (50*30)+(50*30)+(50*30)+(20*30)+(50*30) = 6600 (A large number!)

Why filters help

Filters reduce total number of weights - each “slice” of output volume corresponds to single filter - single set of weights

Filters also allow detection of relationships between pixels

Filters look for features - using the same filter across image makes sense - a feature could be present at any region of the image

Hyperparameters to keep track of

Number of filters (K) (output volume depth)

Stride of filters (S) (how many pixels to skip while moving)

Zero-padding (P) (we'll see this now)

Zero Padding

May want the output volume to have particular spatial arrangement

Zero padding can help.

Add zeros around the edges of the input volume (“pad”)

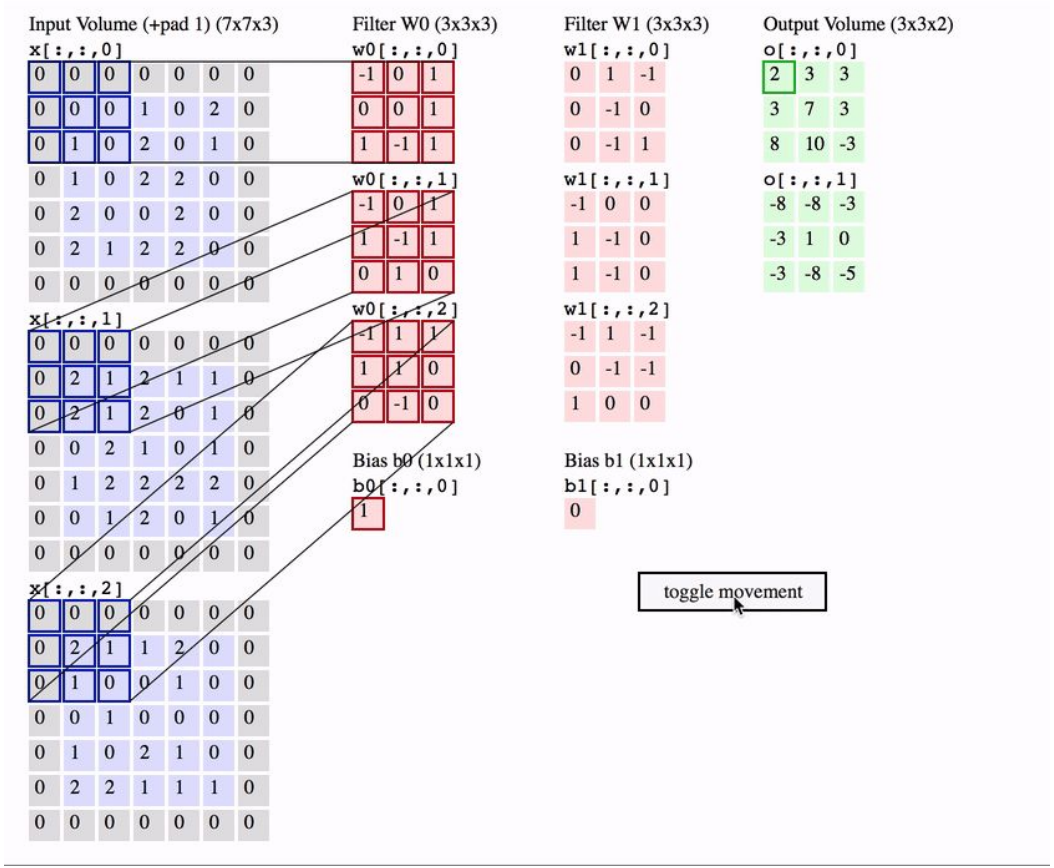
Output volume size is determined by:

$$\text{floor}((W - F + 2P) / S) + 1$$

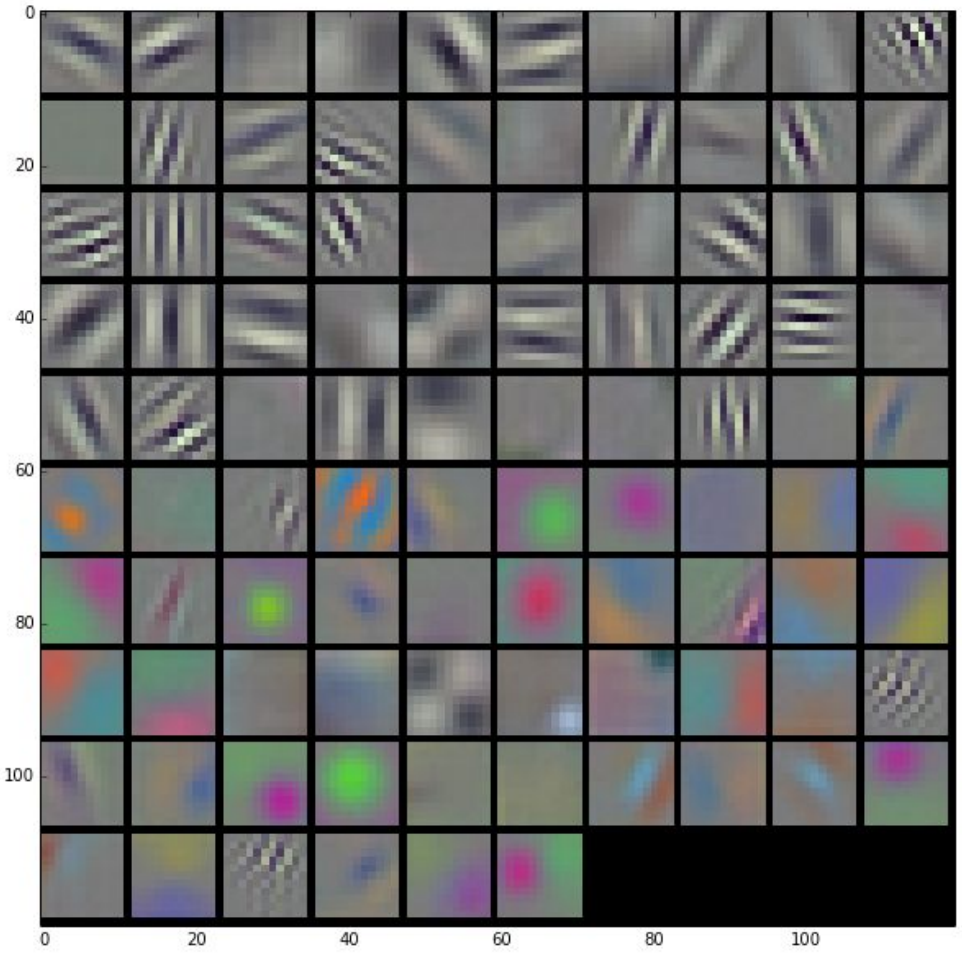
W - Input volume size, F - Filter size, P - Zero Padding

S - Stride

Conv Layer – Visualization (multi filter)



Filters – First layer (ImageNet)



Zero Padding

What is zero padding used for?

“

Questions?

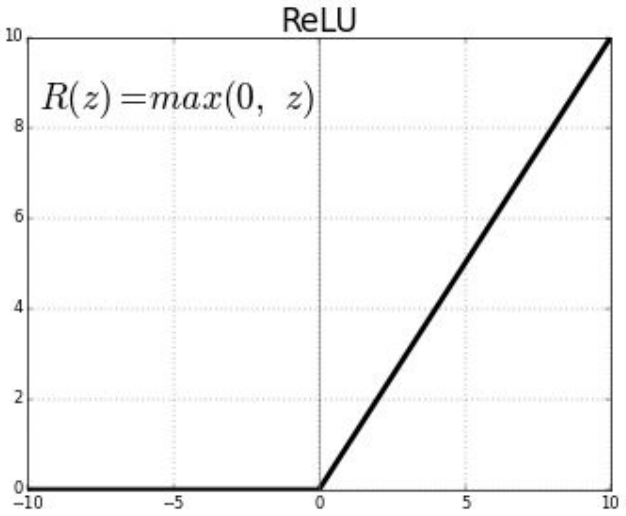
4. Other Layers

Non-linear Layer (ReLU)

What exactly is ReLu and what about its shape is important?

Non-linear Layer (ReLU)

- We apply ReLU to the output of convolutional layers
- Why ReLU?
  - derivative of sigmoid has a small range (0, 0.25]
    - leads to vanishing gradient
  - ReLU gradient values are larger
  - ReLU can “kill” neurons (negative input), creating sparseness



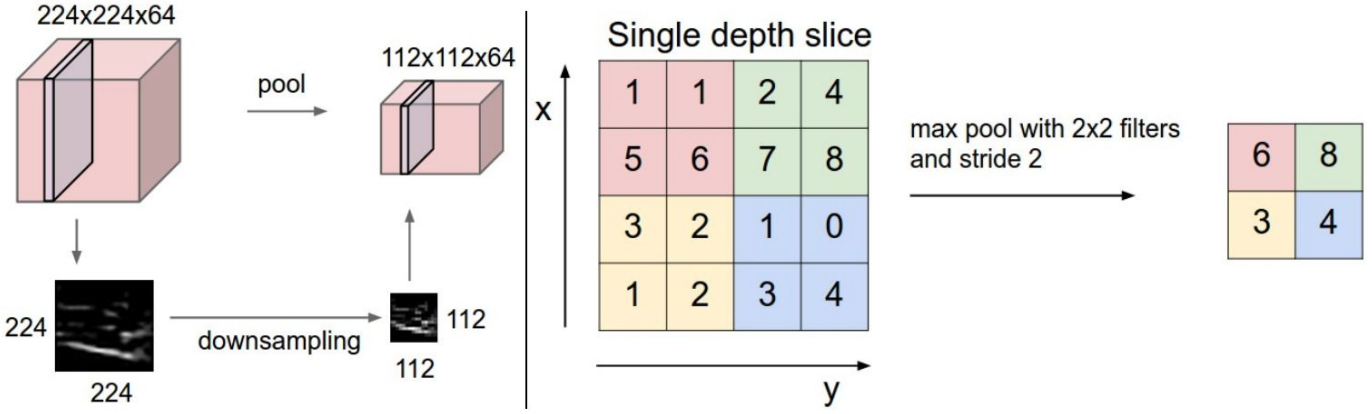
Pooling Layer

Objectives:

1. reduce spatial size of features (height and width not depth)
2. reduce amount of parameters and computations
3. control overfitting

Max Pooling

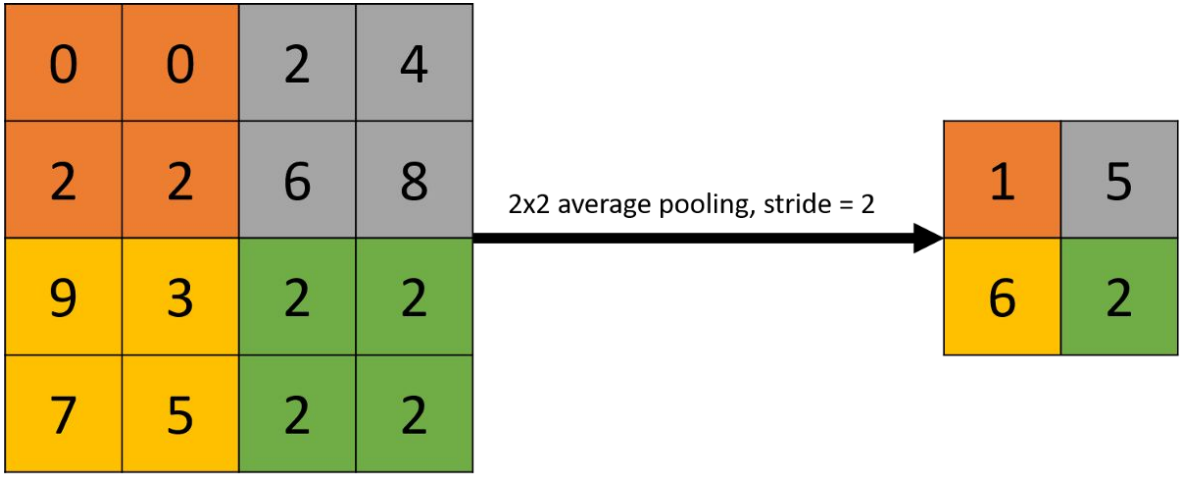
Take the maximum value from each f x f section.



Filter size = 2, stride = 2, downsample by 75%

Average Pooling

Take the average value from each f x f section.



Pooling Layer Common Practices

- Possible parameters for filter (F) and stride (S):
  - F = 2, S = 2
  - F = 3, S = 2 (Overlapping pooling)
- What about padding?
  - uncommon to use padding in the pooling layer
- Of the pooling layers, max pooling generally performs best in practice

Pooling Layer Questions

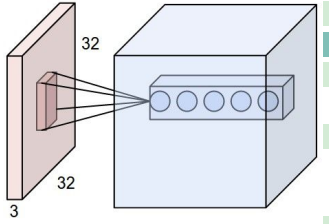
What is the purpose of a pooling layer?

Why might it be okay that we lose information during pooling?

Why do you think pooling is going out of trend?

Making Predictions

- need a way to compute predictions for each class
- For vanilla neural networks we used softmax
  - input is vector where each element corresponds to a class
  - outputs vector of probabilities for each class
- Convolutional layers don't output vectors
  - outputs are 3D (h x w x c)
  - we want to convert that output to a vector

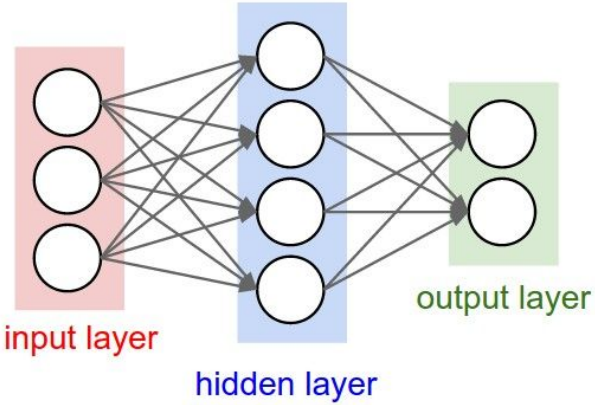


Fully Connected Layer

Fully connected (FC): neurons have full connections to all activations in the previous neural networks

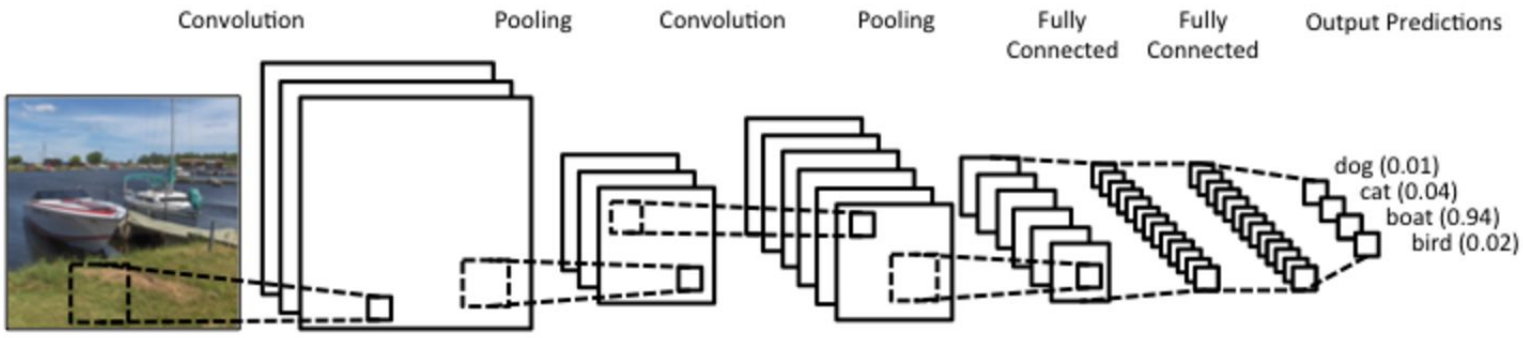
- just like layers in a vanilla neural network

- we “unroll” (flatten) the output of the last layer before the FC layers
- forward propagate the flattened output through the FC layers like in a vanilla neural network



Architecture - General Layer Patterns

Input → (Conv → ReLU) → MaxPool → FC Layers → Softmax



( ) - repeated layer

“

Questions?

Please Fill Out Our Feedback Form!

Other Resources

- Beginner's Guide to Understanding CNNs
- Stanford CS231n: Convolutional Neural Networks for Visual Recognition
- 2017 CS231n Lectures on YouTube
- Stanford UFLDL CNN Tutorial
- CNN Video Tutorial (Brandon Rohrer)
- CNN Chapter from NN and Deep Learning Book

ACM AI

Thank you all for coming!

Coming up next week:
Applications in Convolutional Neural Networks

Join our Facebook group