Lecture Notes for

Neural Networks and Machine Learning

Fully Convolutional Learning

Logistics and Agenda

- Logistics
  - Lab one due soon!
- Agenda
  - Town Hall
  - Paper Presentation
  - Segmentation
    - Semantic (this time)
    - Object (partially this time, maybe)
    - Instance (next time)

Lab One Town Hall

Tamás Görbe @TamasGorbe · 8h

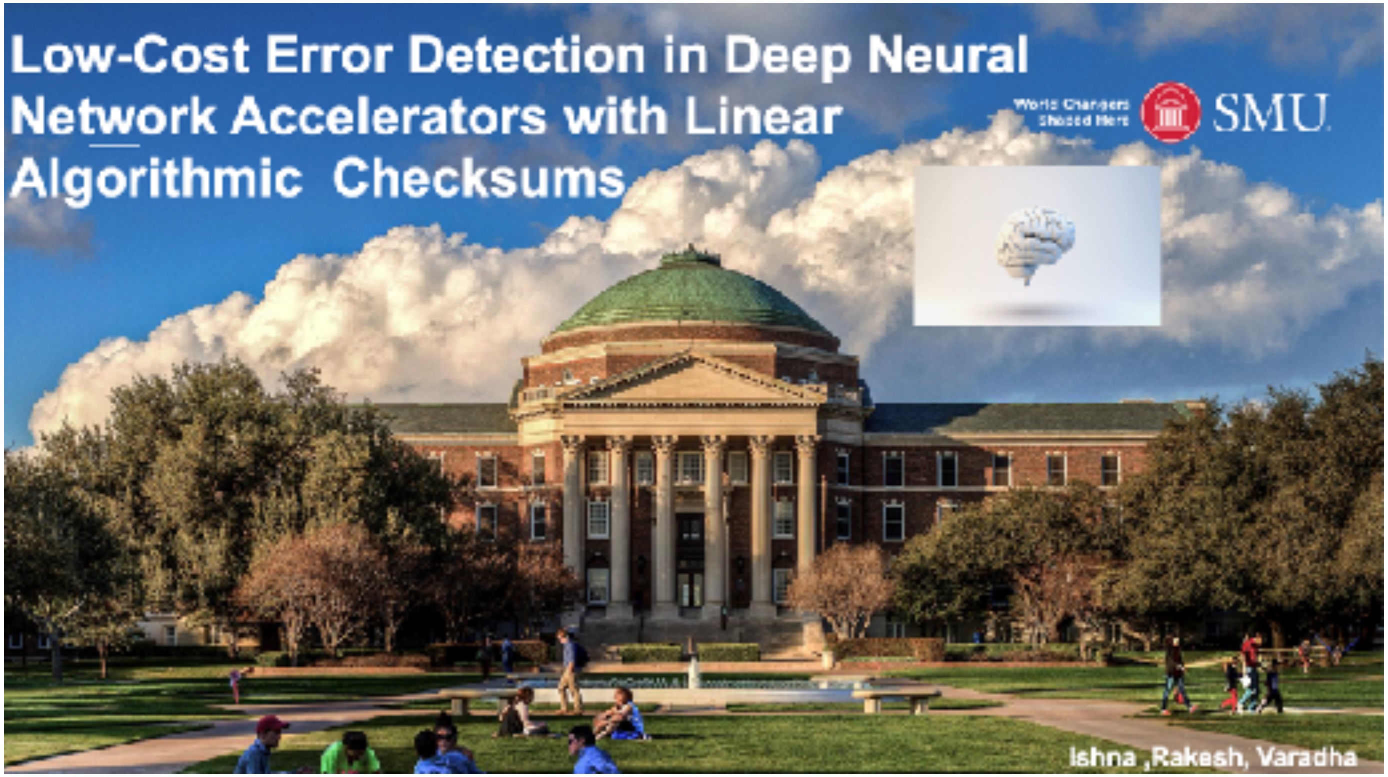
student: how do i become a grad.student?

me: here *hands them a nabla ∇*

∇student

@TamasGorbe

Student Paper Presentation



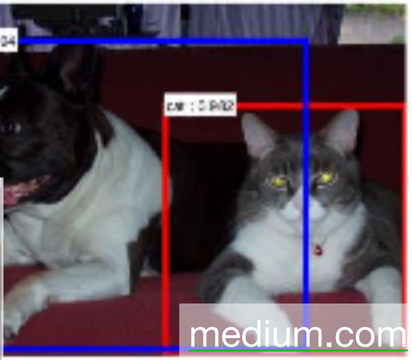
Low-Cost Error Detection in Deep Neural Network Accelerators with Linear Algorithmic Checksums

Ishna ,Rakesh, Varadha

Types of Fully Convolutional Problems

- Semantic Segmentation
- Object Detection
- Instance Segmentation

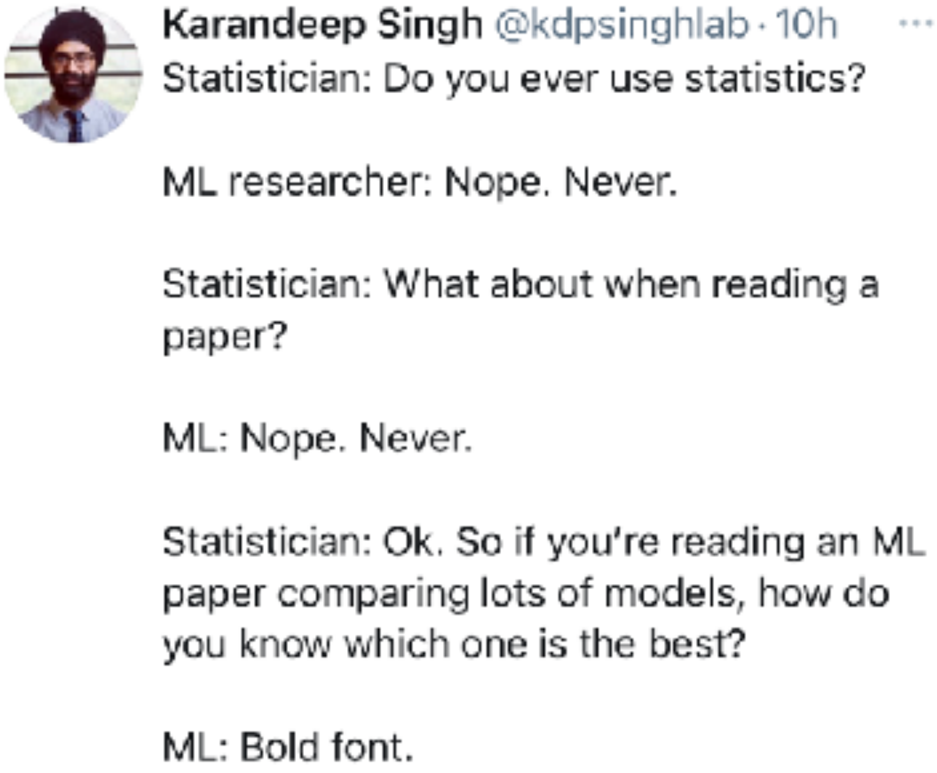
medium.com



medium.com

He e al., Mask r-cnn, 2018

Semantic Segmentation



Karandeep Singh @kdpsinghlab · 10h ···

Statistician: Do you ever use statistics?

ML researcher: Nope. Never.

Statistician: What about when reading a paper?

ML: Nope. Never.

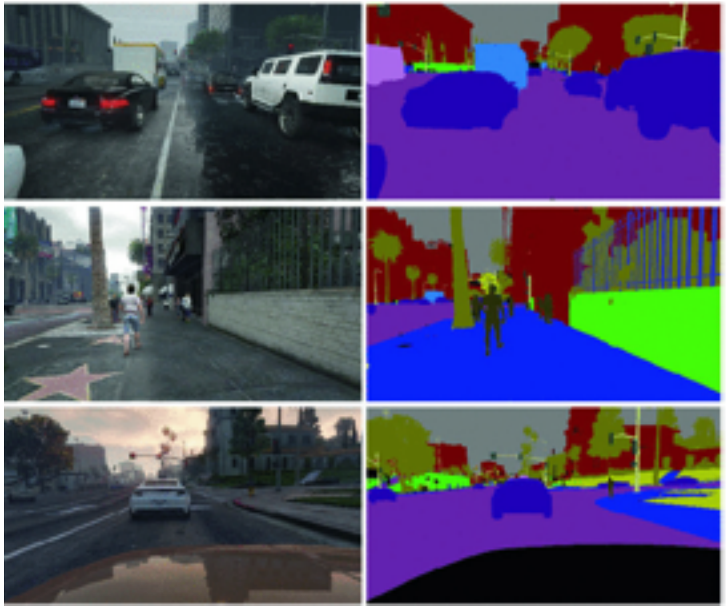
Statistician: Ok. So if you're reading an ML paper comparing lots of models, how do you know which one is the best?

ML: Bold font.

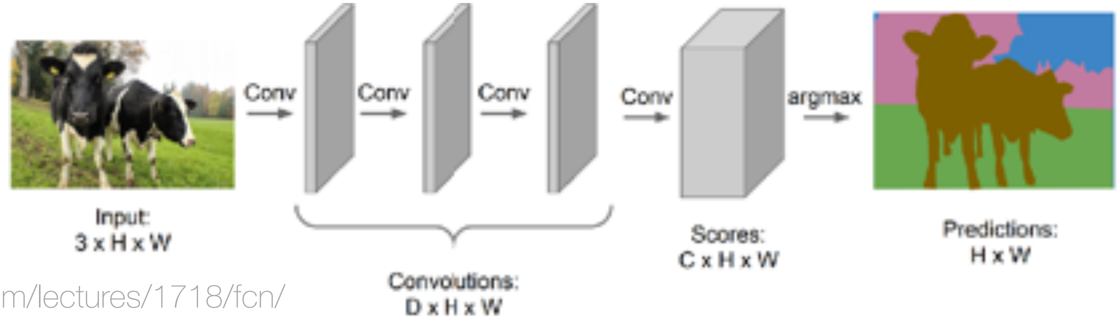
Semantic Segmentation

- Given a set of pixels, classify each pixel according to what instance it belongs

Input



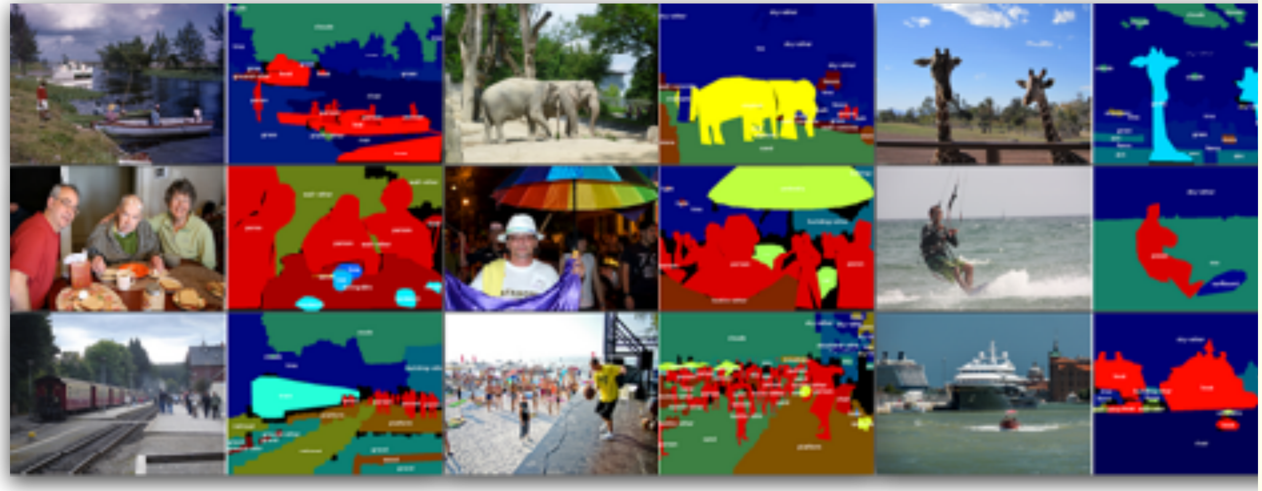
Labels

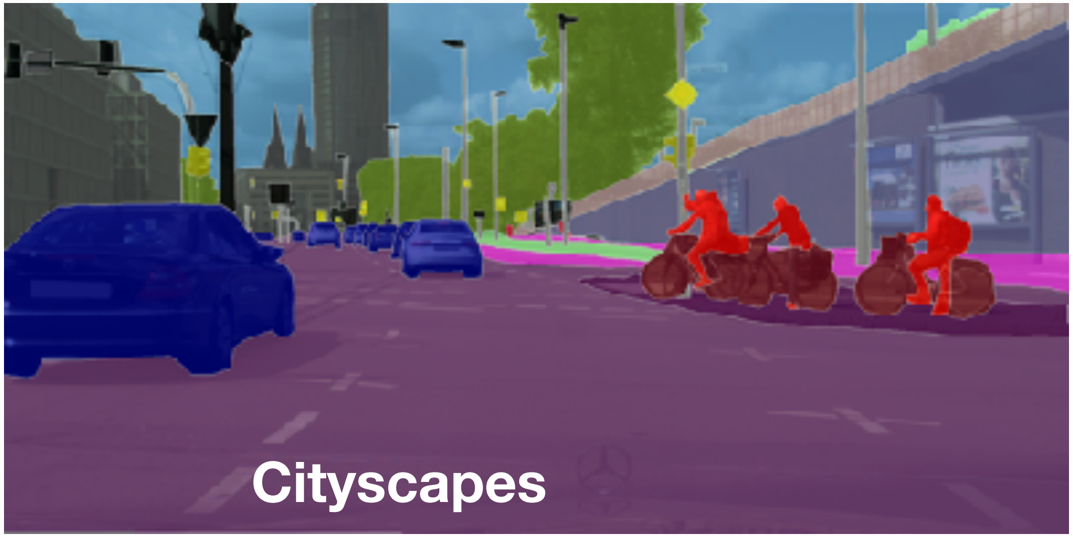


https://tjmachinelearning.com/lectures/1718/fcn/

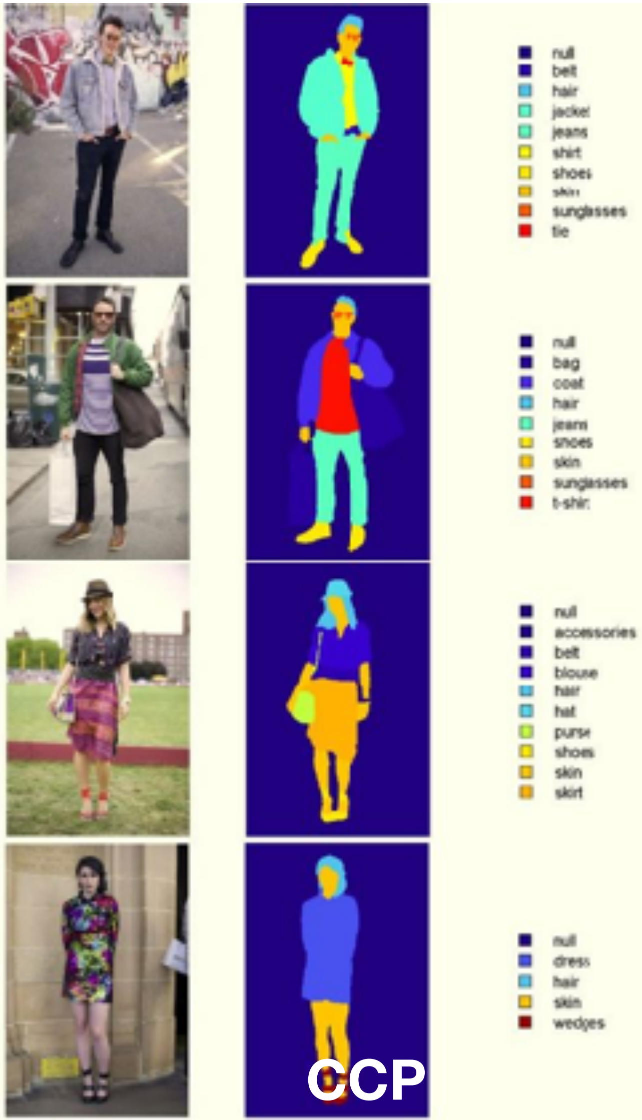
Popular Semantic Segmentation Datasets

COCO http://cocodataset.org/



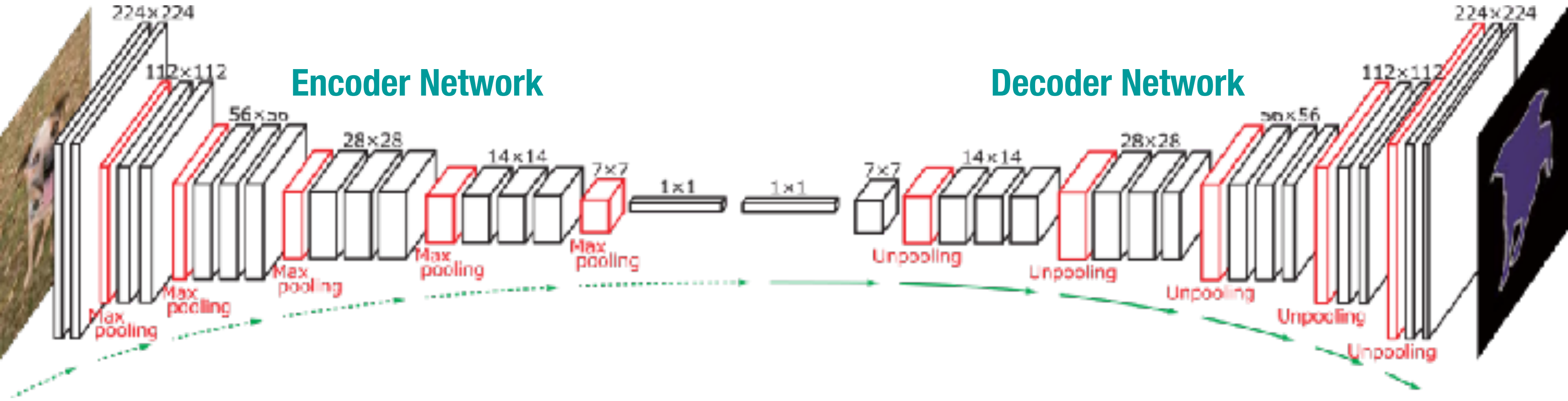


Cityscapes



CCP

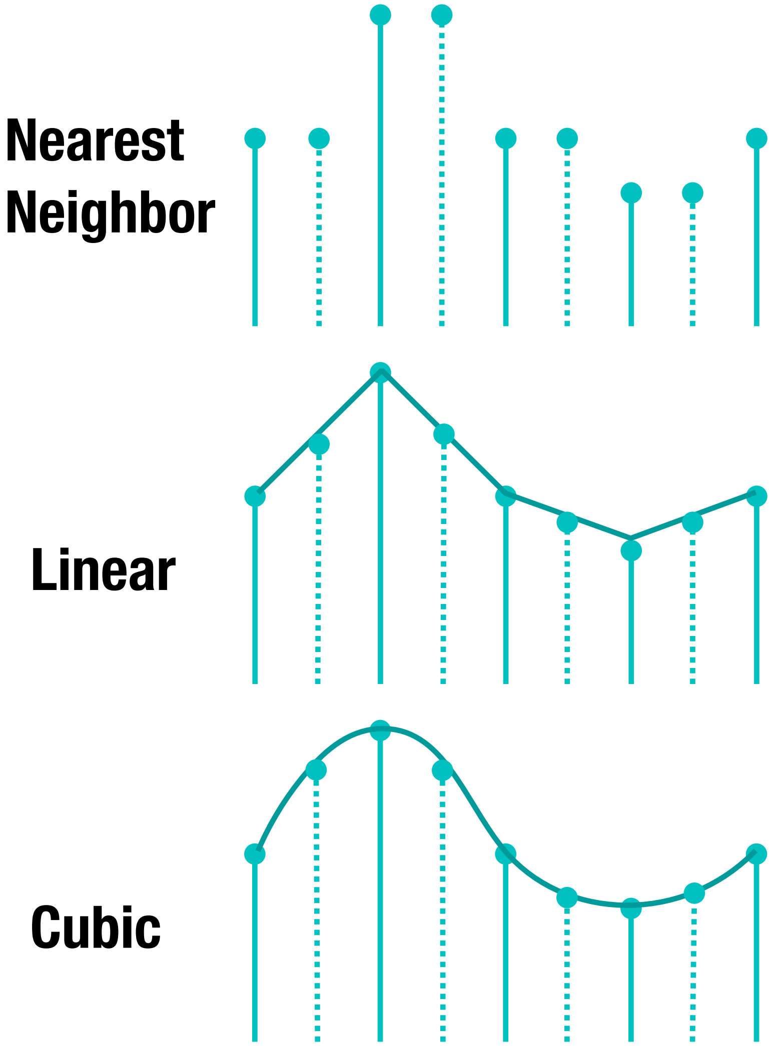
Early Training Methods



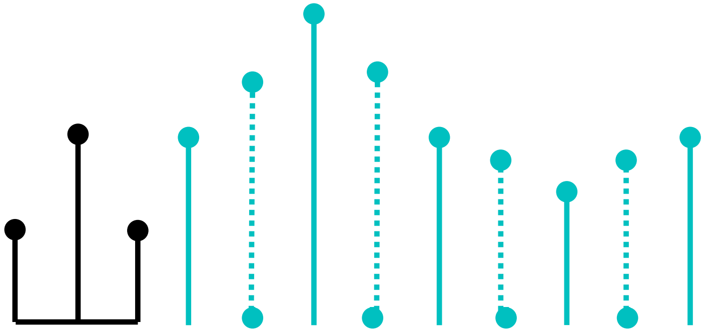
- Init Encoder with traditional CNN (like VGG or DarkNet)
- Freeze encoder and train decoder with segmented image maps
- Unfreeze encoder and fine tune
  - Repeat tuning as needed

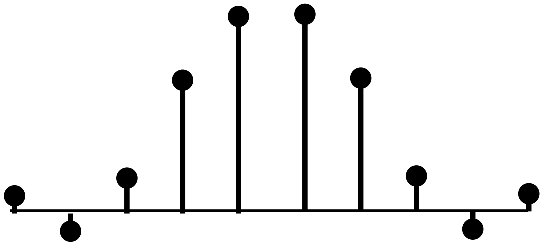
https://tjmachinelearning.com/lectures/1718/fcn/

Aside: Integer Upsampling via Interpolation



All are equivalent to inserting zeros and applying convolutional filter





Aside: Image Upsampling, Integer Factor

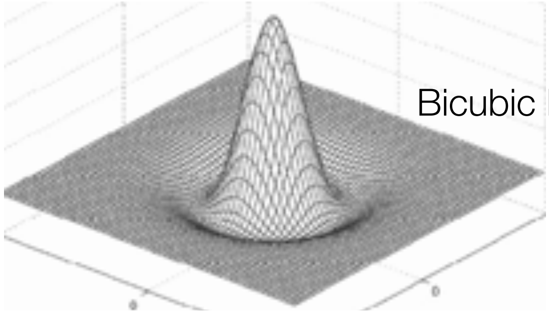
- Insert Zeros
- Convolve

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

1		2		3		4	
5		6		7		8	
9		10		11		12	
13		14		15		16	

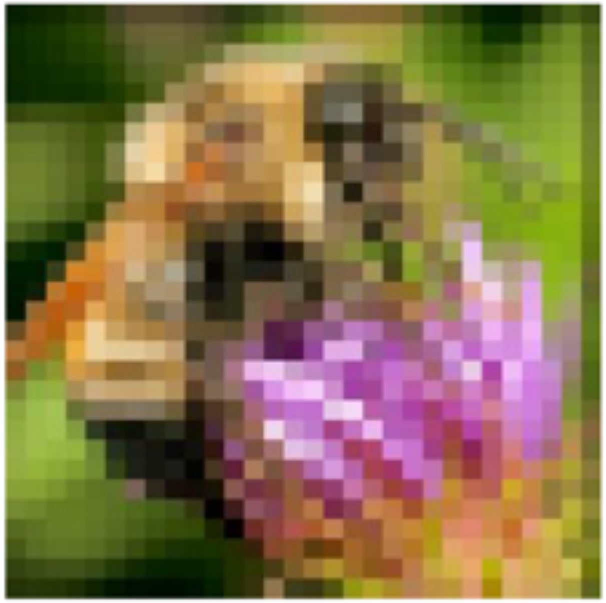
0.25	0.5	0.25
0.5	1	0.5
0.25	0.5	0.25

Bilinear Filtering

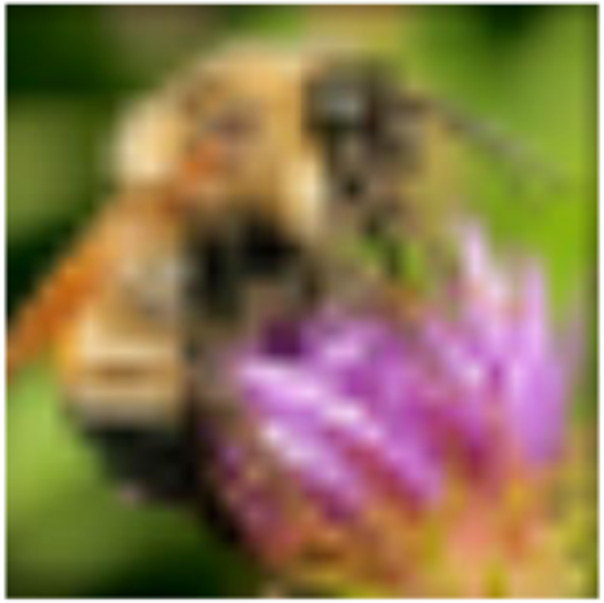


Bicubic Filter

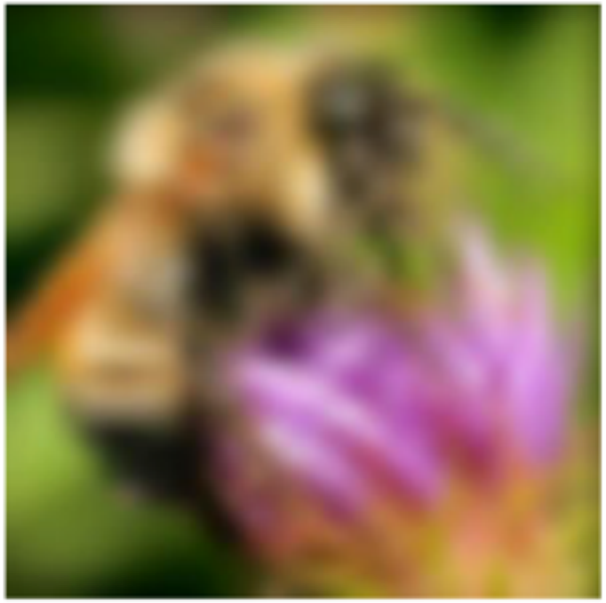
Aside: Image Upsampling, Integer Factor



Nearest Neighbor



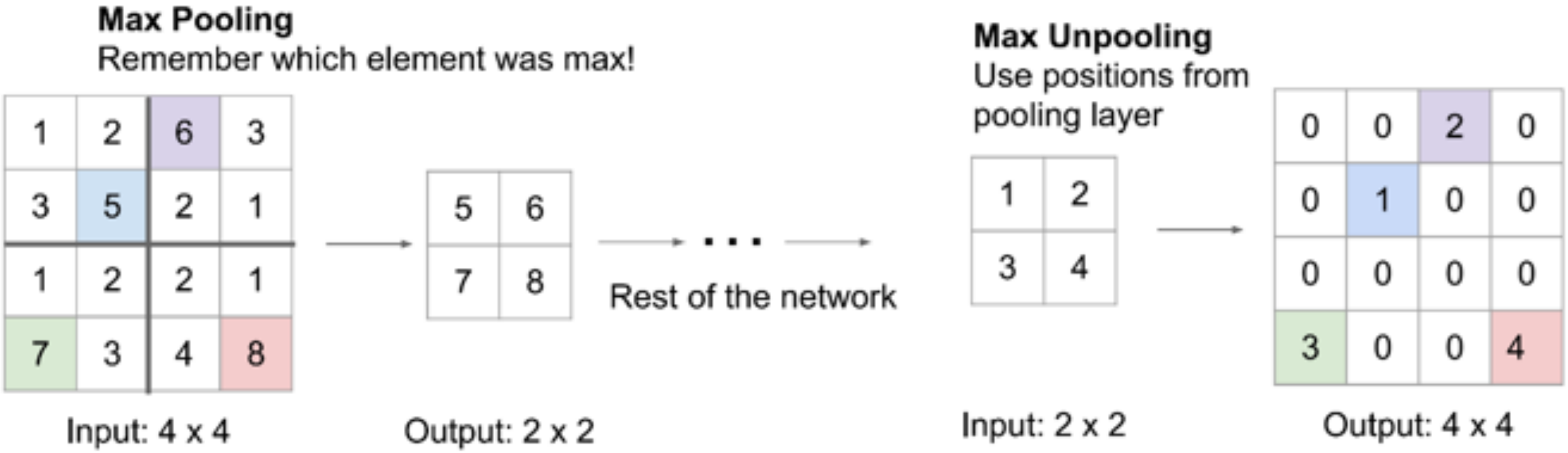
Bilinear



Bicubic

https://www.cs.toronto.edu/~guerzhoy/320/lec/upsampling.pdf

Semantic Segmentation



Max Pooling
Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

5	6
7	8

Output: 2 x 2

Rest of the network

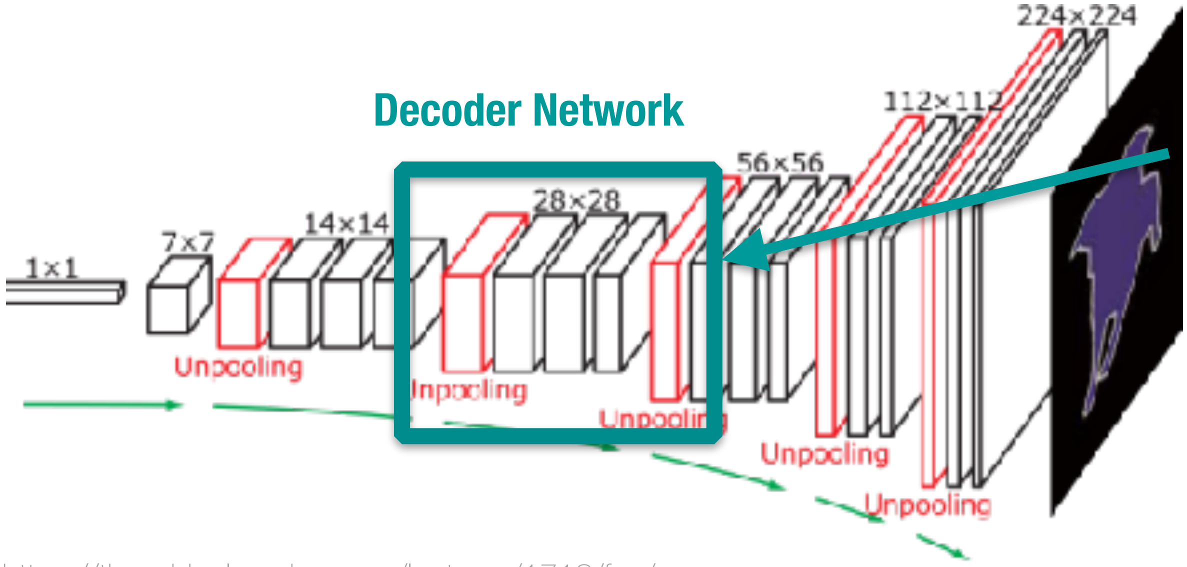
Max Unpooling
Use positions from pooling layer

1	2
3	4

Input: 2 x 2

0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Output: 4 x 4



https://tjmachinelearning.com/lectures/1718/fcn/

Some knucklehead started calling this **deconvolution**. If you use that term in this class, **you fail**.

This is unpooling and then convolution, but **now the interpolation filters are learned**!!

What about transpose convolution?

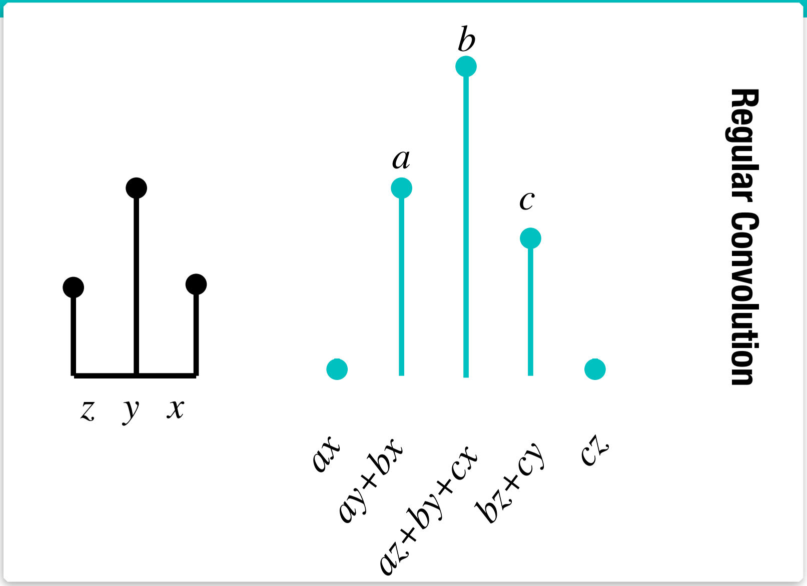
Convolution as Matrix Multiplication

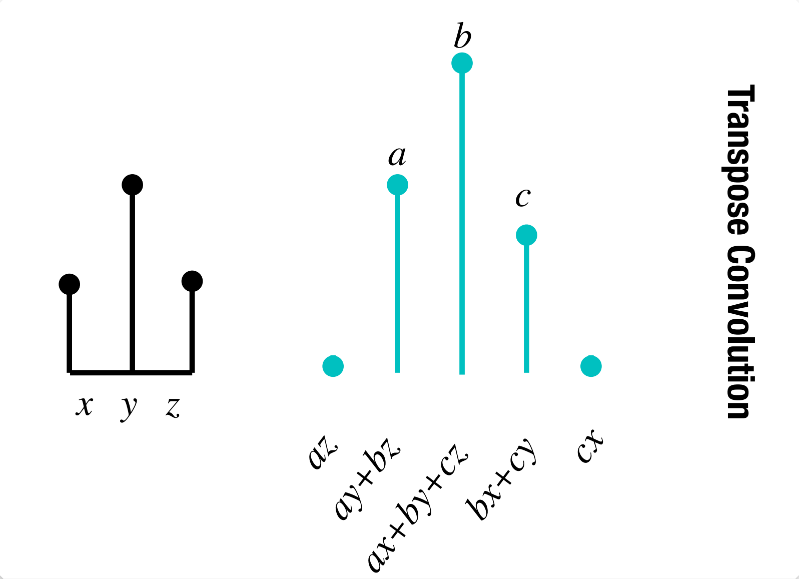
$$\begin{bmatrix} y & x & 0 & 0 & 0 \\ z & y & x & 0 & 0 \\ 0 & z & y & x & 0 \\ 0 & 0 & z & y & x \\ 0 & 0 & 0 & z & y \end{bmatrix} \times \begin{bmatrix} 0 \\ a \\ b \\ c \\ 0 \end{bmatrix} = \begin{bmatrix} ax \\ ay+bx \\ az+by+cx \\ bz+cy \\ cz \end{bmatrix}$$

Transpose

$$\begin{bmatrix} y & z & 0 & 0 & 0 \\ x & y & z & 0 & 0 \\ 0 & x & y & z & 0 \\ 0 & 0 & x & y & z \\ 0 & 0 & 0 & x & y \end{bmatrix} \times \begin{bmatrix} 0 \\ a \\ b \\ c \\ 0 \end{bmatrix} = \begin{bmatrix} az \\ ay+bz \\ ax+by+cz \\ bx+cy \\ cx \end{bmatrix}$$

like convolving with “reversed coefficients”





Transpose Convolution: Strides

Strided Convolution as Matrix Multiplication

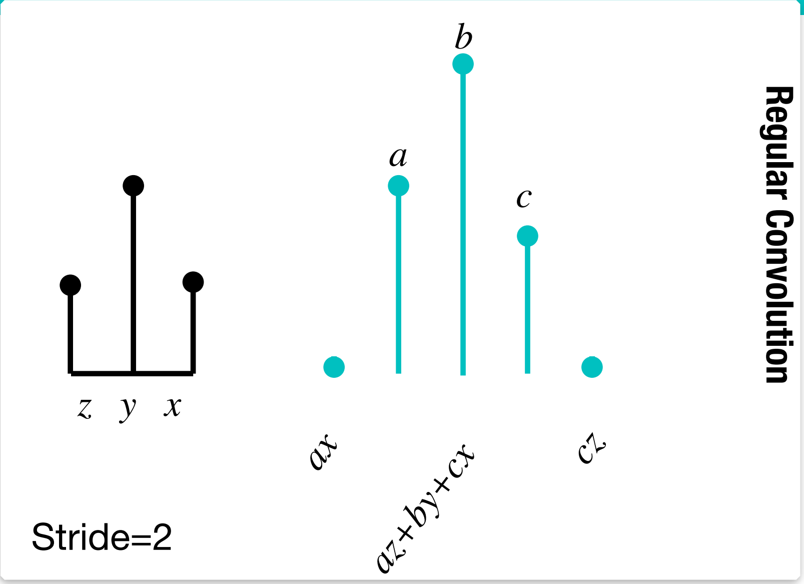
$$\begin{bmatrix} y & x & 0 & 0 & 0 \\ 0 & z & y & x & 0 \\ 0 & 0 & 0 & z & y \end{bmatrix} \times \begin{bmatrix} 0 \\ a \\ b \\ c \\ 0 \end{bmatrix} = \begin{bmatrix} ax \\ az+by+cx \\ cz \end{bmatrix}$$

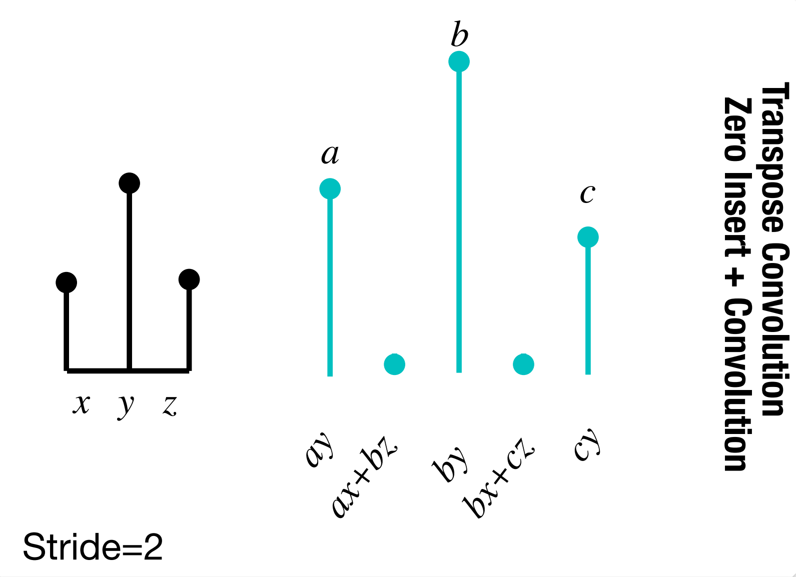
↓

Transpose

↓

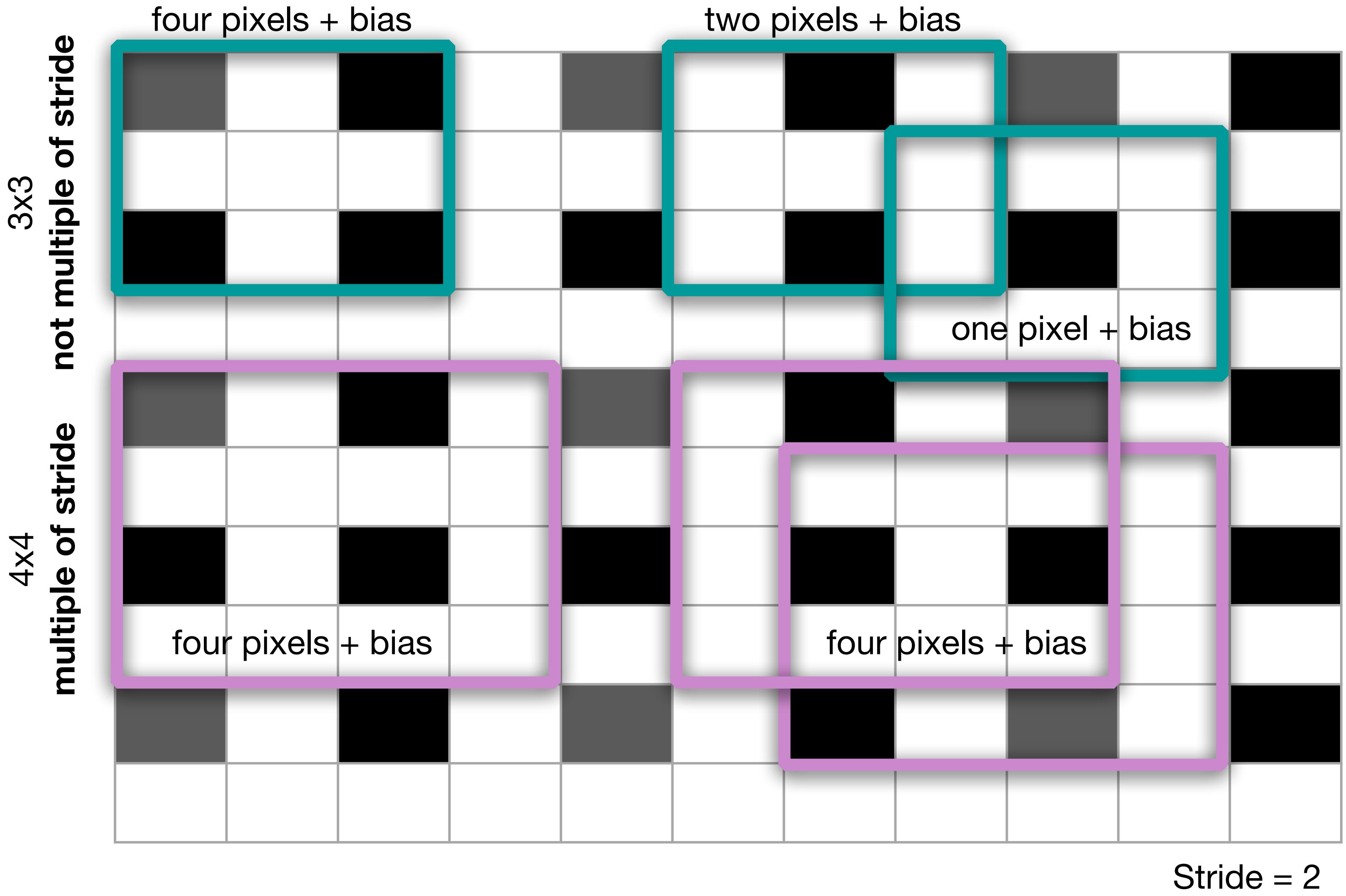
$$\begin{bmatrix} y & 0 & 0 \\ x & z & 0 \\ 0 & y & 0 \\ 0 & x & z \\ 0 & 0 & y \end{bmatrix} \times \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} ay \\ ax+bz \\ by \\ bx+cz \\ cy \end{bmatrix}$$





Aside: Convolution after zero insertion

- Kernel size should be a symmetric multiple of the stride



Bias needs to account for both when different numbers of pixels overlap with the kernel

Multiple of stride ensures that same number of active pixels overlap the kernel.