Topics:

- Advanced Architectures (Segmentation, Detection)
- Calibration (Fairness/Bias)

CS 4803-DL / 7643-A

ZSOLT KIRA

- **Assignment 3 out**
  - Due **March 14th 11:59pm EST.**

- **Projects**
  - Released assignments; please **reach out** to your groups to discuss team formation
    - Note: Some may have already found groups, etc. Note that it doesn't **have** to be 4 members so you can go with smaller. You can also converge on high-level topic and then reach out on piazza looking for members.
  - Rubric/description, project proposal instructions, FB projects released
  - Project proposal due **March 22nd**

Topics: You can choose any topic you'd like for the project, but here are some suggestions

- **Facebook projects** - For these projects you will have FB mentors that you will interact with through forums and office hours,
- **Additional ideas**

Project description and rubrics: Group_Project_Description.pdf

Instructions for Project Proposal:

Here's what you and your teams need to do:

1. ONE person on the team submit a post with your project proposal
2. Others on the team reply to this post that they are part of the team.

This is going to be public for the entire class, feel free to check out other proposals. Of course do not plagiarize content from any one else's proposals or from any references, (having the same reference as another project is ok but you should do your own research).

What goes in a project proposal?

1. Team Name
2. Is this a Facebook project?
3. Project Title
4. Project summary (4-5+ sentences). Fill in your problem and background/motivation (why do you want to solve it? Why is it interesting?). This should provide some detail (don't just say "I'll be working on object detection")
5. What you will do (Approach, 4-5+ sentences) - Be specific about what you will implement and what existing code you will use. Describe what you actually plan to implement or the experiments you might try, etc. Again, provide sufficient information describing exactly what you'll do. One of the key things to note is that just downloading code and running it on a dataset is not sufficient for a description or a project! Some thorough implementation, analysis, theory, etc. has to be done for the project.
6. Resources / Related Work & Papers (4-5+ sentences). What is the state of art for this problem? Note that it is perfectly fine for this project to implement approaches that already exist. This part should show you've done some research about what approaches exist.
7. Datasets (Provide a Link to the dataset). This is crucial! Deep learning is data-driven, so what datasets you use is crucial. One of the key things is to make sure you don't try to create and especially annotate your own data! Otherwise the project will be taken over by this.
8. List your Group members.
9. Are you looking for more members?

Each member of your group will then reply to this post to confirm they are part of the project.

An example Project Proposal:

Team: Next Move

Facebook Project: Yes

Project Title: Motion Prediction

Project Summary:

The ability to forecast human motion is useful for a myriad of applications including robotics, self driving cars, and animation. Typically we consider this a generative modeling task, where given a seed motion sequence, a network learns to generate/synthesize a sequence of plausible human poses. This task has seen much progress for shorter horizon forecasting through traditional sequence modeling techniques; however longer horizons suffer from pose collapse. This project aims to explore recent approaches that can better capture long term dependencies and generate longer horizon sequences.

Approach:

- Based on our preliminary research, there are multiple approaches to address 3D Motion Prediction problem. We want to start by collecting and analyzing varying approaches; e.g. Encoder-Recurrent-Decoder (ERD), GCN, Spatio-Temporal Transformer. We expect to reproduce [1] and baseline other approaches.
- As a stretch goal, we want to explore possible directions to improve these papers. One avenue is to augment the data to provide multiple views of the same motion and ensure prediction consistency.
- Another stretch goal is to come up with a new metric and loss terms (e.g. incorporating physical constraints) to improve benchmarks.

Resources/Related Work:

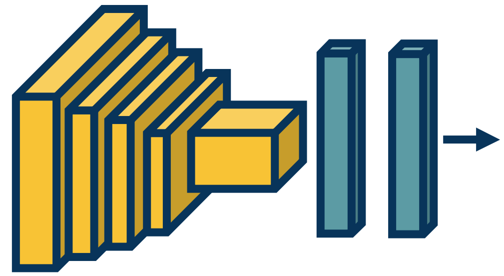
[1] "A Spatio-temporal Transformer for 3D HumanMotion Prediction", Aksan et al.
[2] "Recurrent Network Models for Human Dynamics", Fragkiadaki et al.
[3] "Learning Dynamic Relationships for 3D Human Motion Prediction", Cui et al.
[4] "Convolutional Sequence to Sequence Model for Human Dynamics", Zhang et al
[5] "Attention is all you need", Vaswani et al.
[6] "On human motion prediction using recurrent neural networks", Martinez et al.
[7] "Structured Prediction Helps 3D Human Motion Modelling", Aksan et al.
[8] "Learning Trajectory Dependencies for Human Motion Prediction", Mao et al.
[9] "AMASS: Archive of Motion Capture as Surface Shapes", Mahmood et al.

Datasets:
AMASS https://amass.is.tue.mpg.de/

Team Members:
Eren Jaeger
Armin Arlert
Mikasa Ackerman

Looking for more members:
No

Given a **trained** model, we'd like to understand what it learned.



Weights

Fei-Fei Li, Justin Johnson, Serena Yeung, from CS 231n

Zeiler & Fergus, 2014

Activations

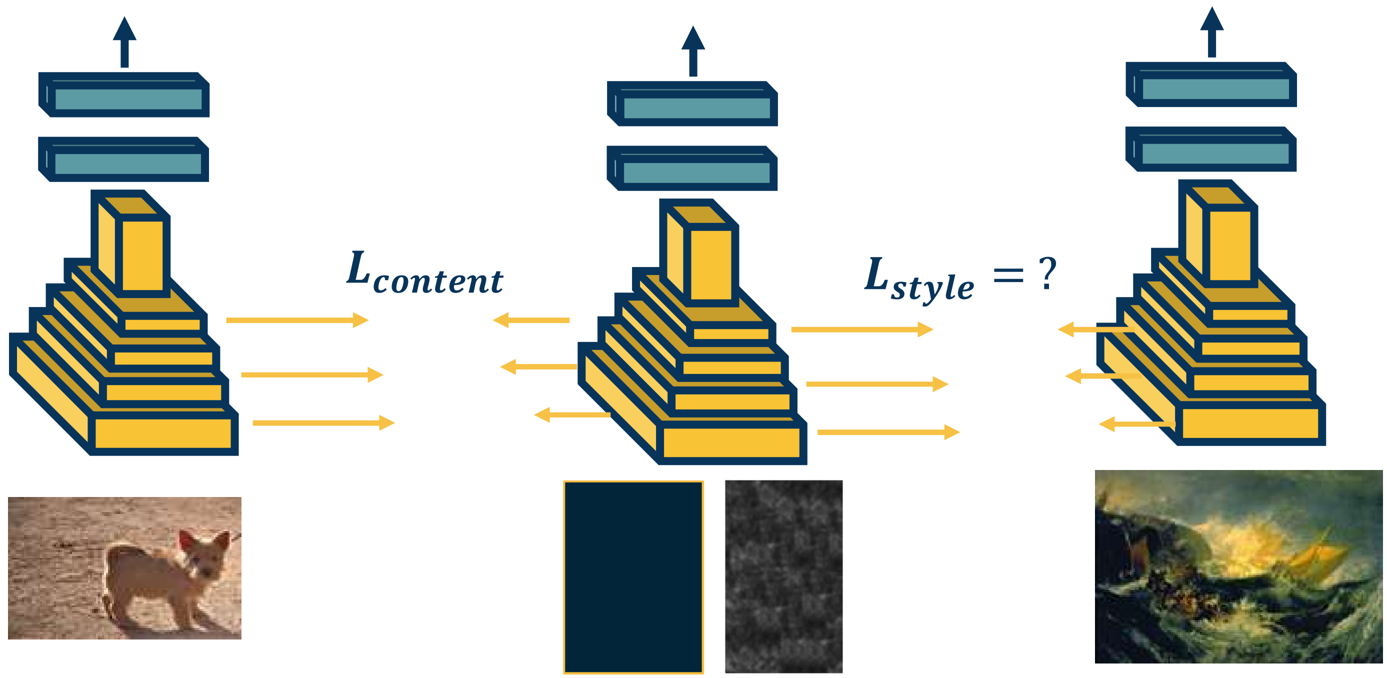
Gradients

Simonyan et al, 2013

Robustness

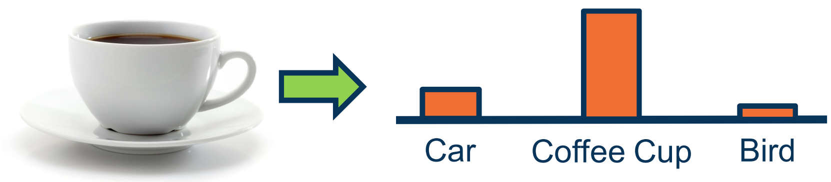
Hendrycks & Dietterich, 2019

- **Idea:** Can we have the *content* of one image and *texture* (style) of another image?
  - **Yes!**



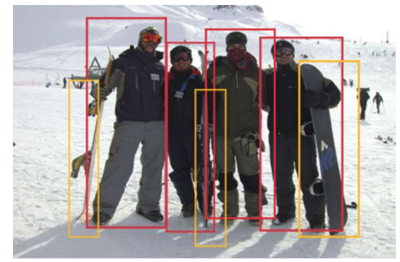
Replicating Content and Style

Image Segmentation Networks



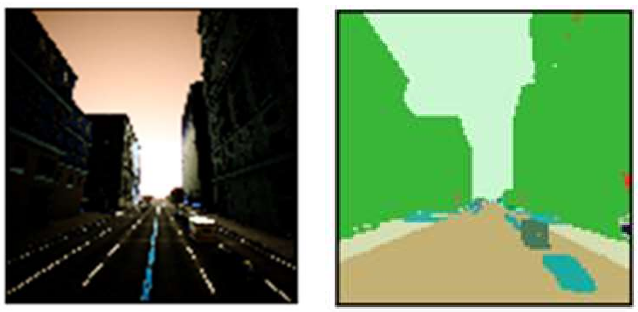
Classification

(Class distribution per image)



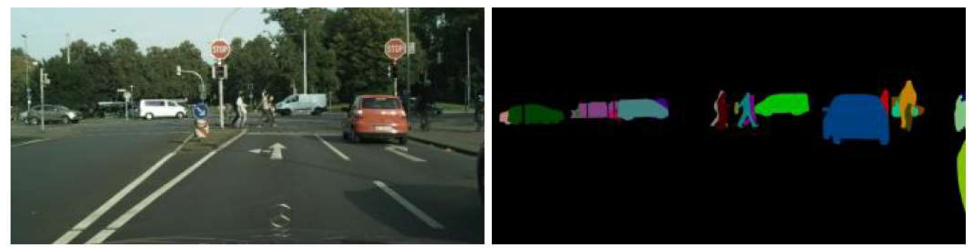
Object Detection

(List of bounding boxes with class distribution per box)



Semantic Segmentation

(Class distribution per pixel)



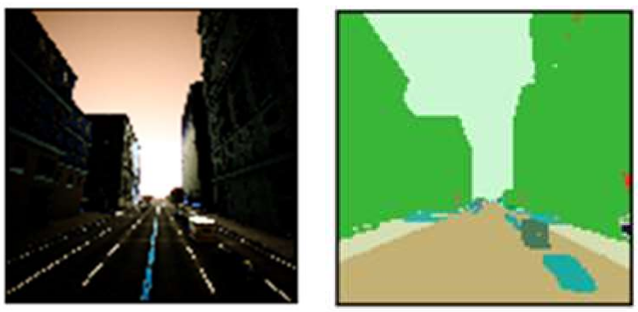
Instance Segmentation

(Class distribution per pixel with unique ID)

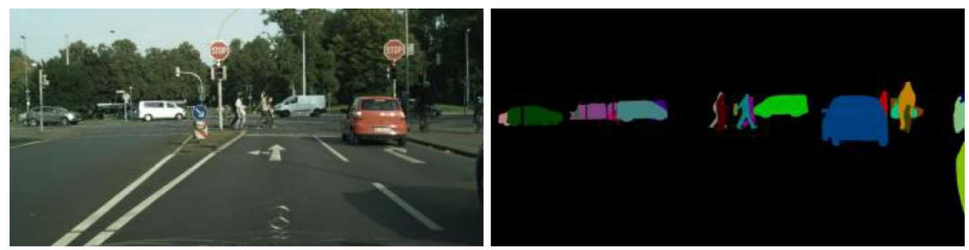
Computer Vision Tasks

Given an image, output another image

- Each output contains class distribution per pixel
- More generally an image-to-image problem

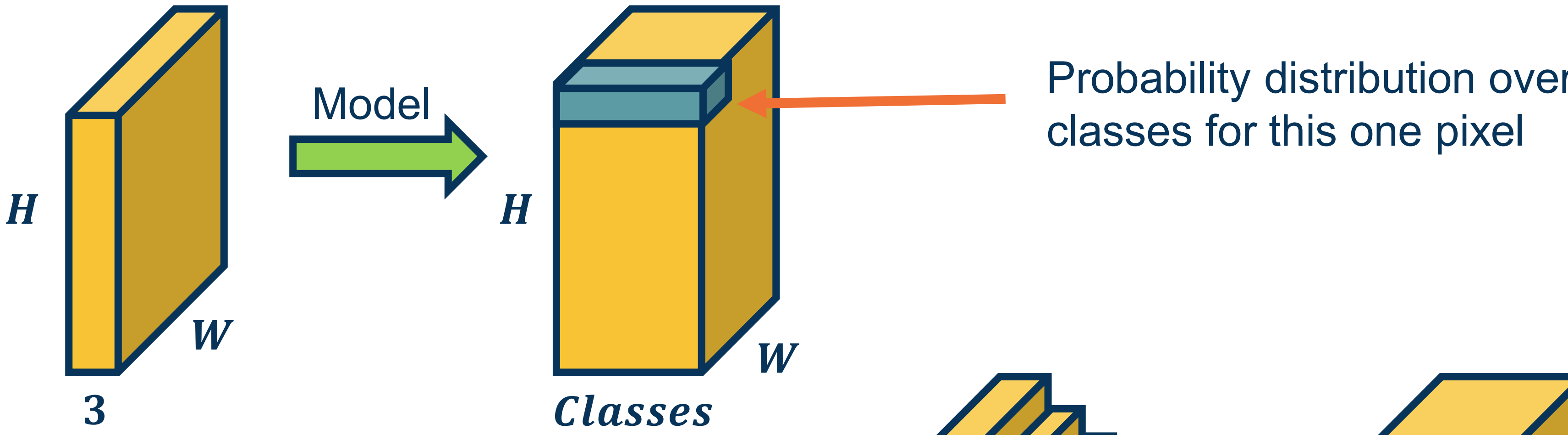


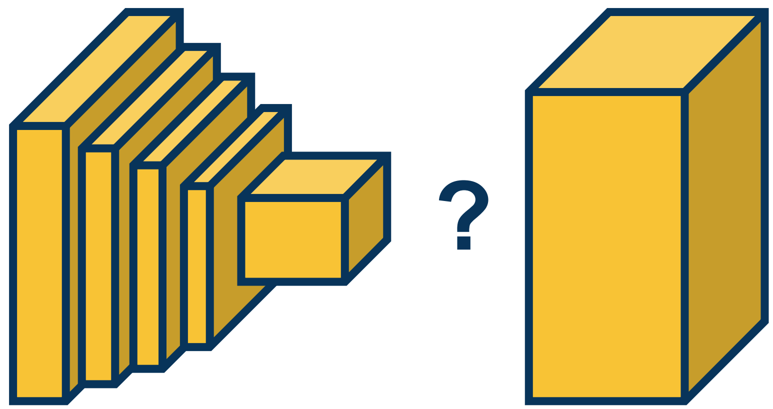
Semantic Segmentation
(Class distribution per pixel)



Instance Segmentation
(Class distribution per pixel with unique ID)

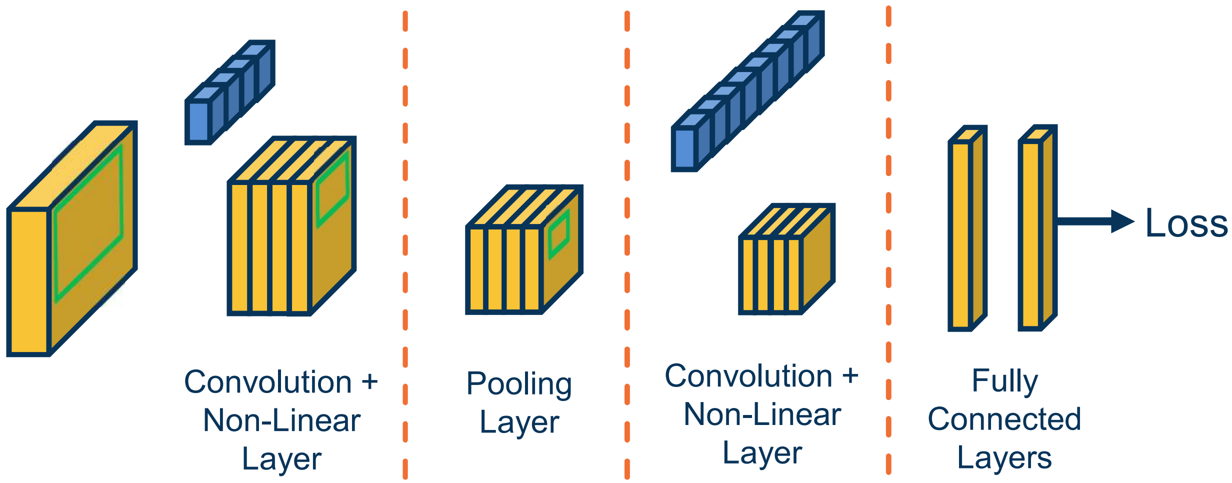
Segmentation Tasks





Input & Output

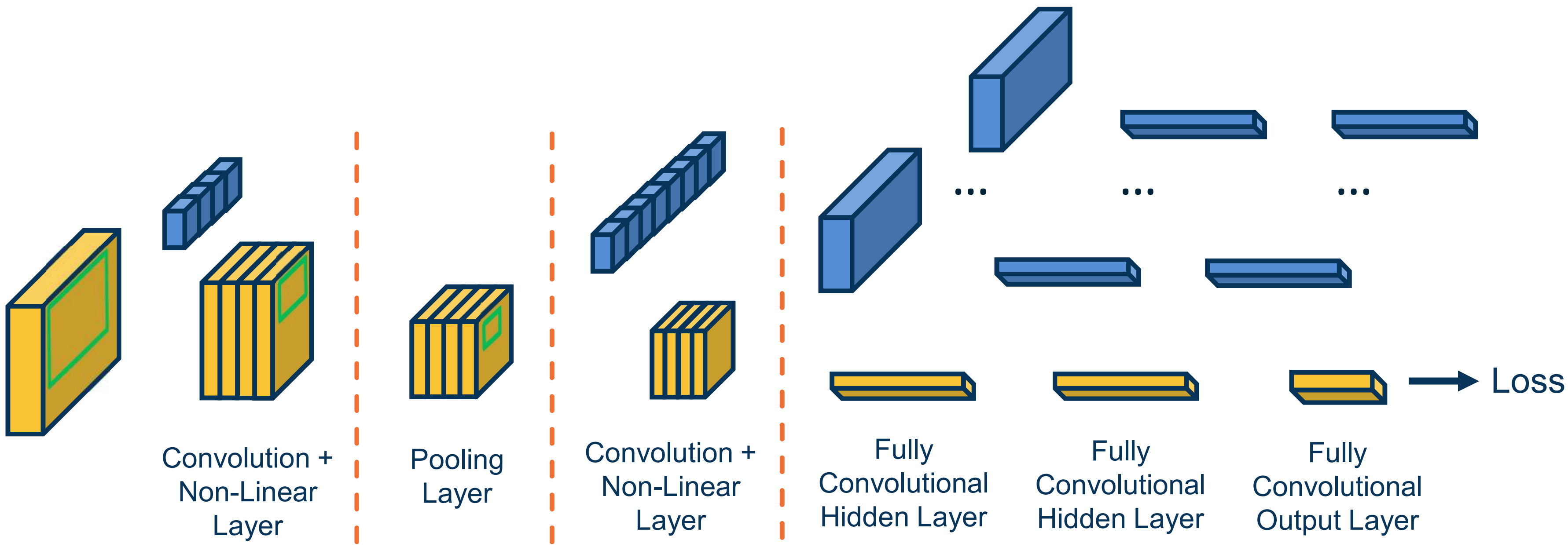
Georgia Tech



Fully connected layers no longer explicitly retain spatial information (though the network can still learn to do so)

Idea: Convert fully connected layer to convolution!

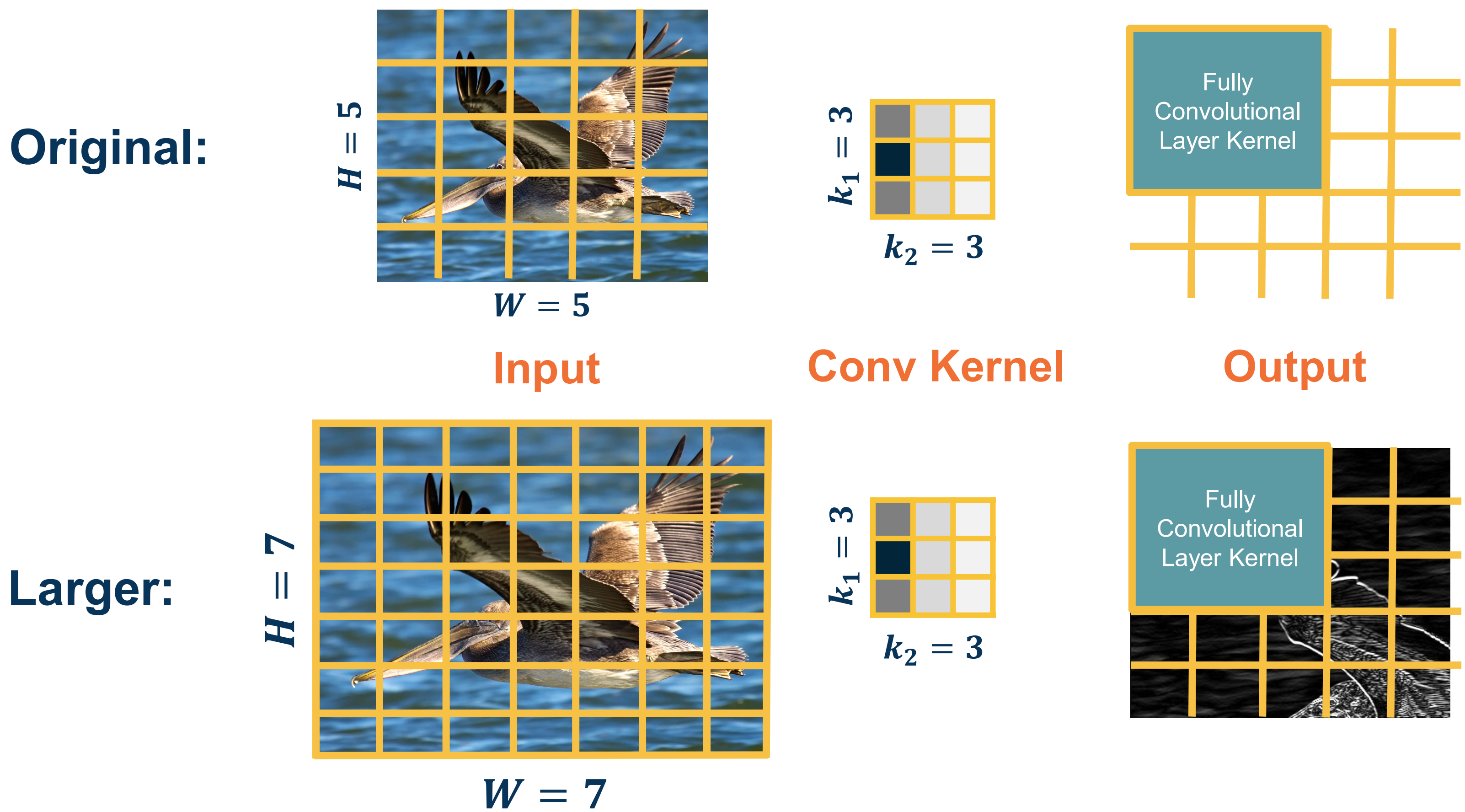
Idea 1: Fully-Convolutional Network



Each kernel has the size of entire input! (output is 1 scalar)

- This is equivalent to Wx+b!
- We have one kernel per output node

Converting FC Layers to Conv Layers



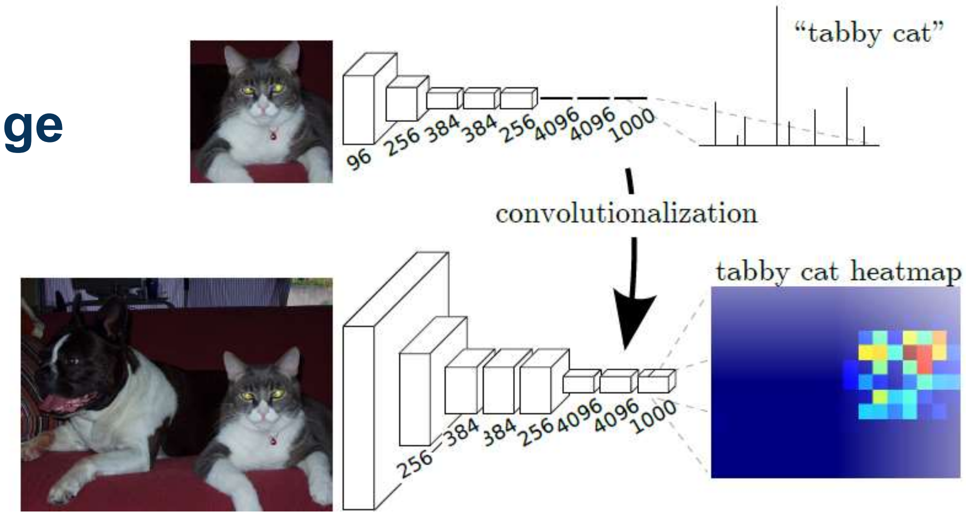
Same Kernel, Larger Input

Why does this matter?

- We can stride the "fully connected" classifier across larger inputs!
- Convolutions work on arbitrary input sizes (because of striding)

Original sized image

Larger Image

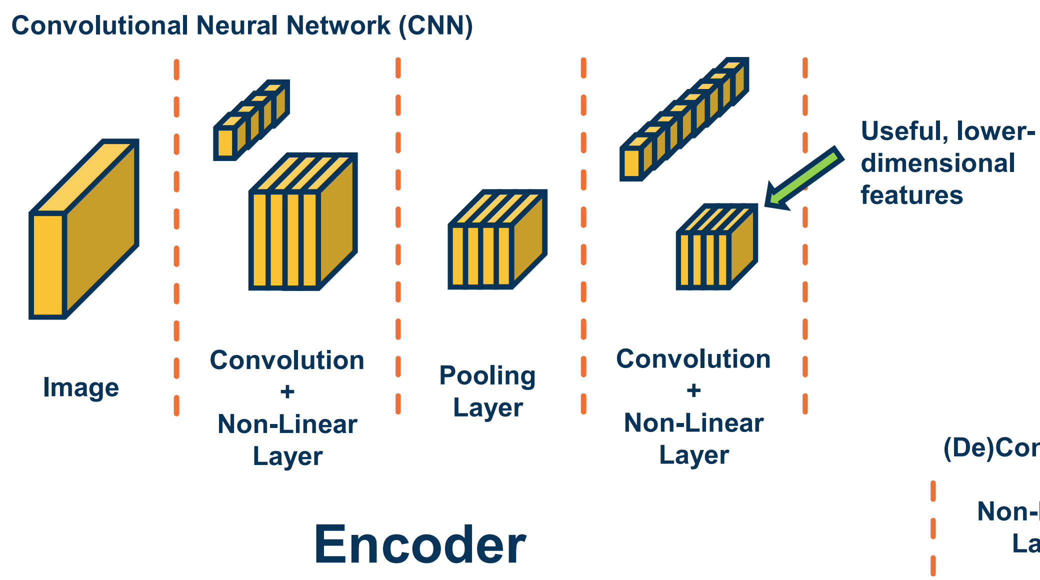


Larger Output Size!

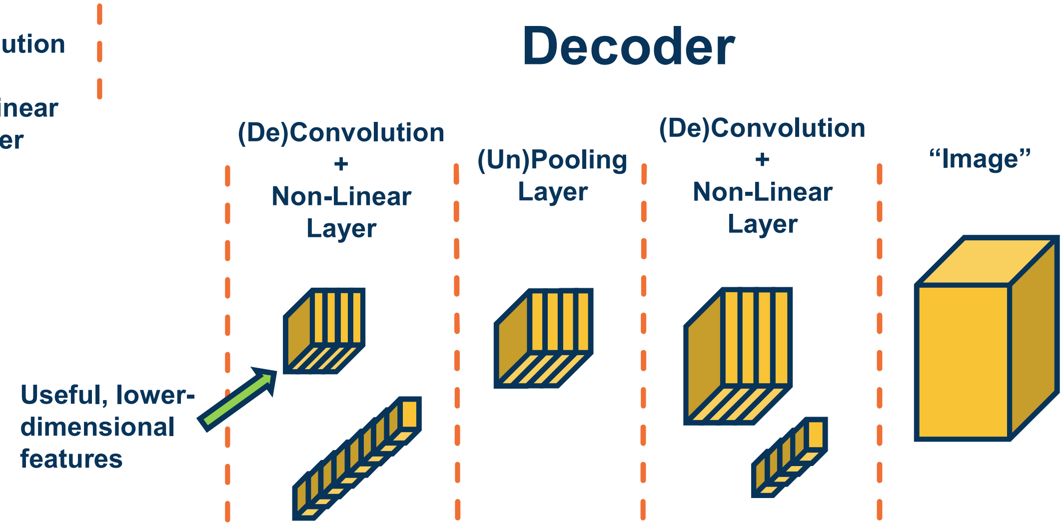
Larger Output Maps

Long, et al., "Fully Convolutional Networks for Semantic Segmentation", 2015

Inputting Larger Images



We can develop learnable or non-learnable upsampling layers!



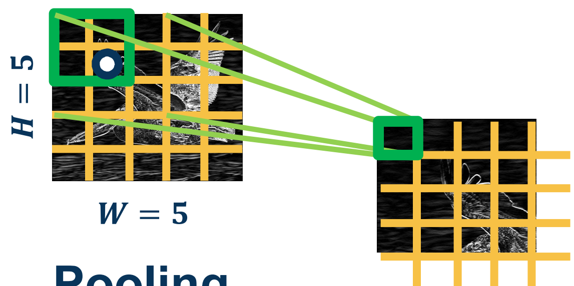
Idea 2: "De"Convolution and UnPooling

Georgia Tech

Example : Max pooling

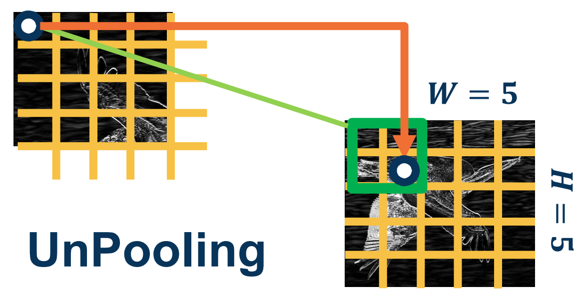
- Stride window across image but perform per-patch **max operation**

$$X(0{:}1, 0{:}1) = \begin{bmatrix} 100 & 150 \\ 100 & 200 \end{bmatrix} \Rightarrow \max(0{:}1, 0{:}1) = 200$$



Pooling

Copy value to position chosen as max in encoder, fill reset of this window with zeros



UnPooling

Idea: Remember max elements in encoder! Copy value from equivalent position, rest are zeros

Max Unpooling

$$X = \begin{bmatrix} 120 & 150 & 120 \\ 100 & 50 & 110 \\ 25 & 25 & 10 \end{bmatrix} \xrightarrow[\text{2x2 max pool}]{} Y = \begin{bmatrix} 150 & 150 \\ 100 & 110 \end{bmatrix}$$

Encoder

Decoder

$$X = \begin{bmatrix} 300 & 450 \\ 100 & 250 \end{bmatrix} \xrightarrow{\text{2x2 max unpool}} Y = \begin{bmatrix} 0 & 300 & - \\ 0 & 0 & - \\ - & - & - \end{bmatrix}$$

Max Unpooling Example (one window)

Max Unpooling Example

Encoder

$$X_{\mathrm{enc}} = \begin{bmatrix} 120 & 150 & 120 \\ 100 & 50 & 110 \\ 25 & 25 & 10 \end{bmatrix} \xrightarrow{\text{2x2 max pool}} Y_{\mathrm{enc}} = \begin{bmatrix} 150 & 150 \\ 100 & 110 \end{bmatrix}$$

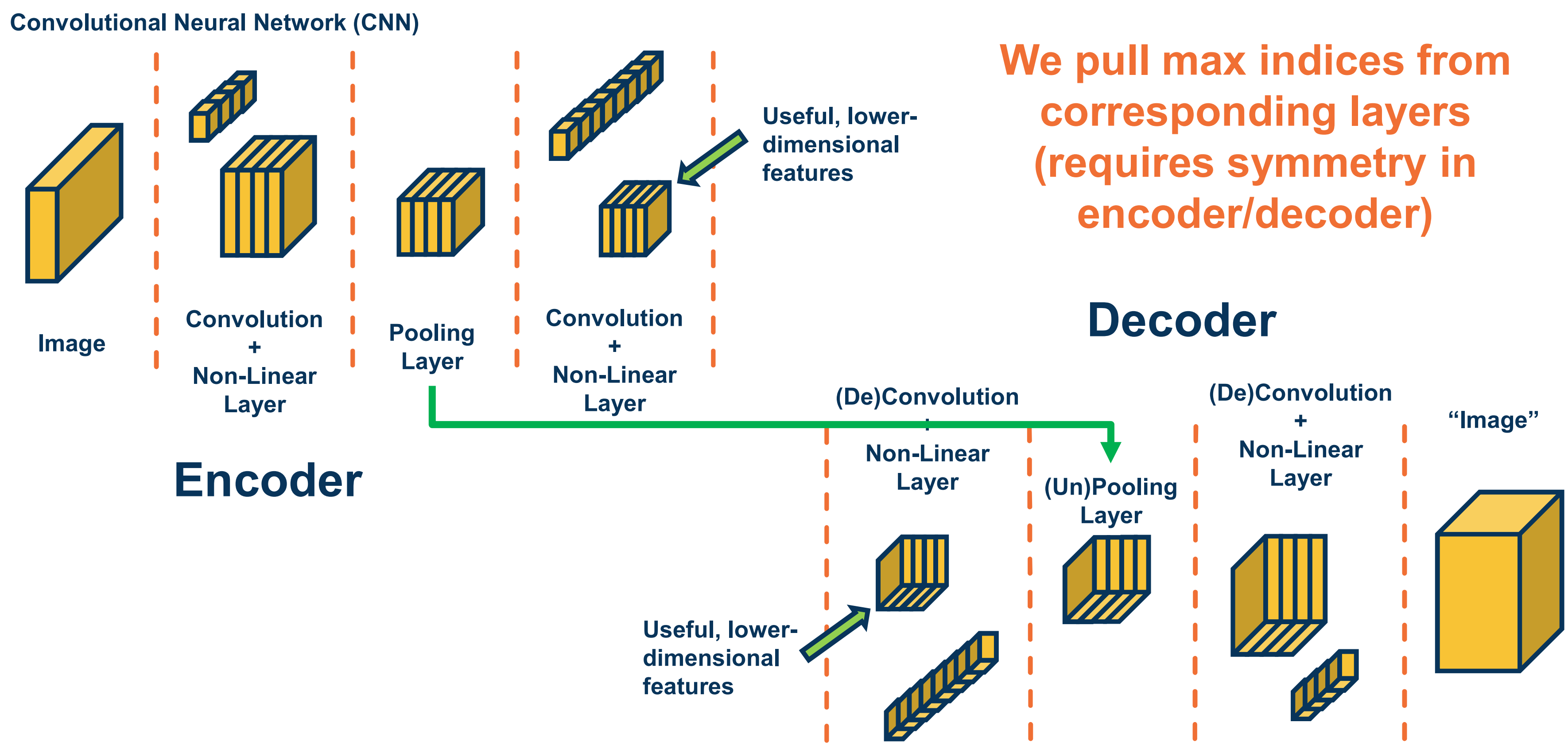
Contributions from multiple windows are summed

Decoder

$$X_{\mathrm{dec}} = \begin{bmatrix} 300 & 450 \\ 100 & 250 \end{bmatrix} \xrightarrow{\text{2x2 max unpool}} Y_{\mathrm{dec}} = \begin{bmatrix} 0 & 300+450 & 0 \\ 100 & 0 & 250 \\ 0 & 0 & 0 \end{bmatrix}$$

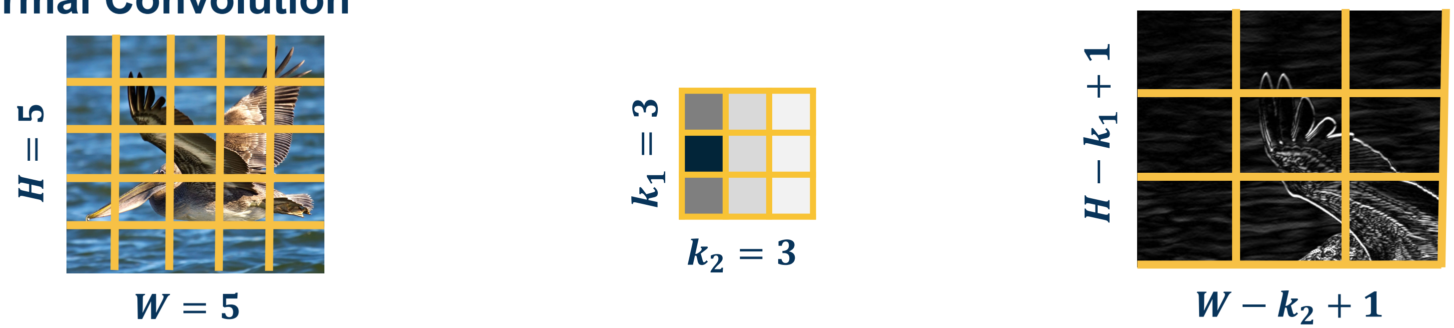
Symmetry in Encoder/Decoder

We pull max indices from corresponding layers (requires symmetry in encoder/decoder)



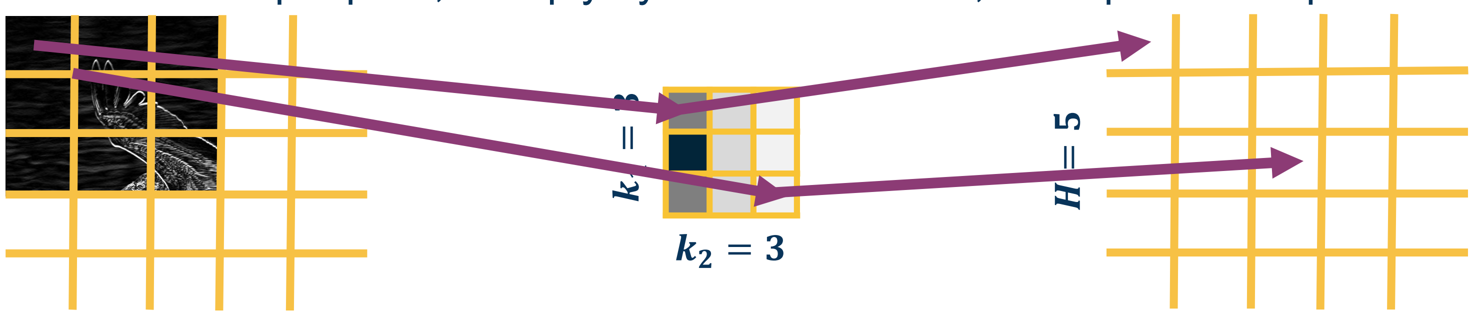
How can we *upsample* using convolutions and learnable kernel?

Normal Convolution



Transposed Convolution (also known as "deconvolution", fractionally strided conv)

Idea: Take each input pixel, multiply by learnable kernel, "stamp" it on output



"De"Convolution (Transposed Convolution)

$$X = \begin{bmatrix} 120 & 150 & 120 \\ 100 & 50 & 110 \\ 25 & 25 & 10 \end{bmatrix} \qquad K = \begin{bmatrix} 1 & -1 \\ 2 & -2 \end{bmatrix}$$

Contributions from multiple windows are summed

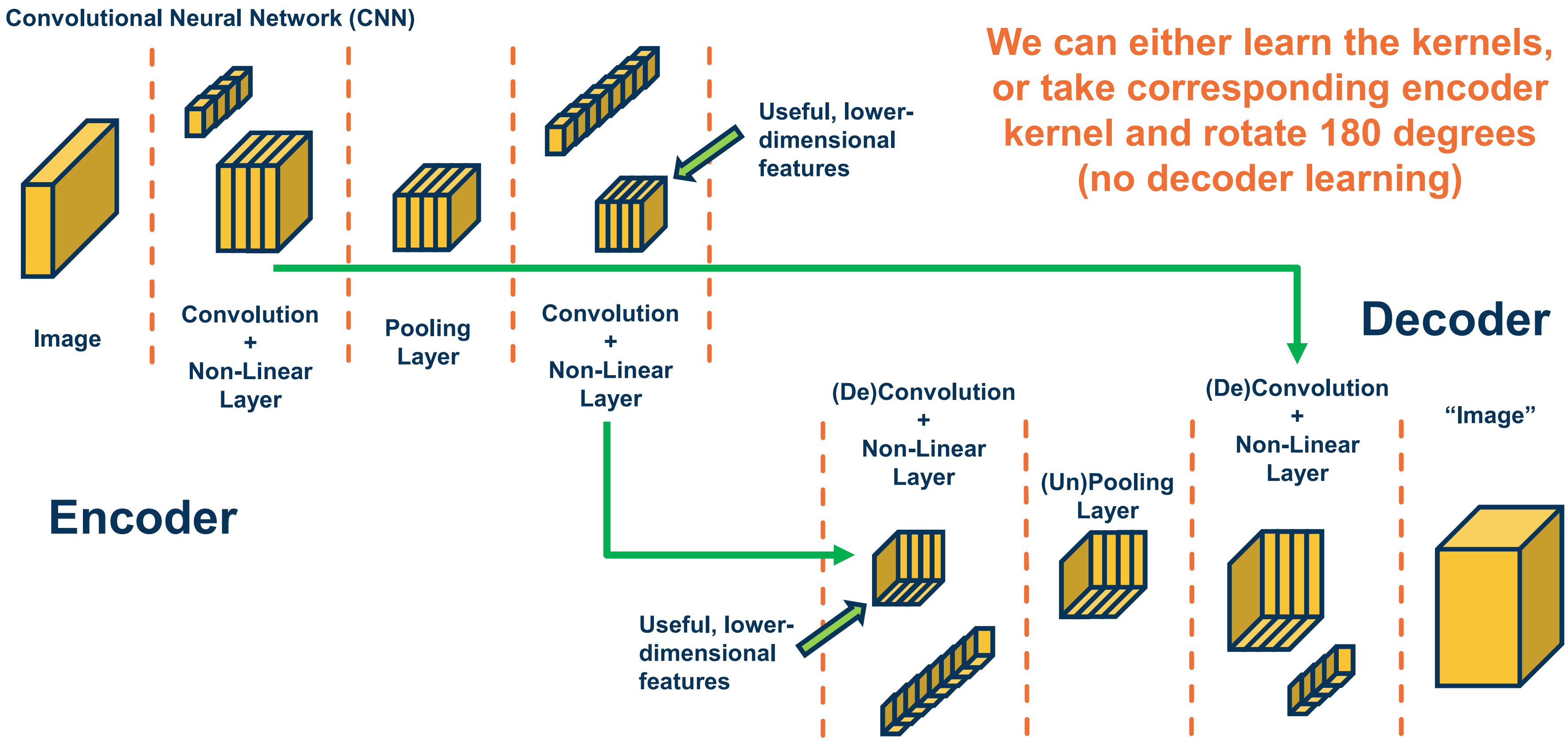
$$\begin{bmatrix} 120 & -120 & 0 & 0 \\ 240 & -240 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Incorporate
X(0,0)

$$\begin{bmatrix} 120 & -120+150 & -150 & 0 \\ 240 & -240+300 & -300 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Incorporate
X(1,0)

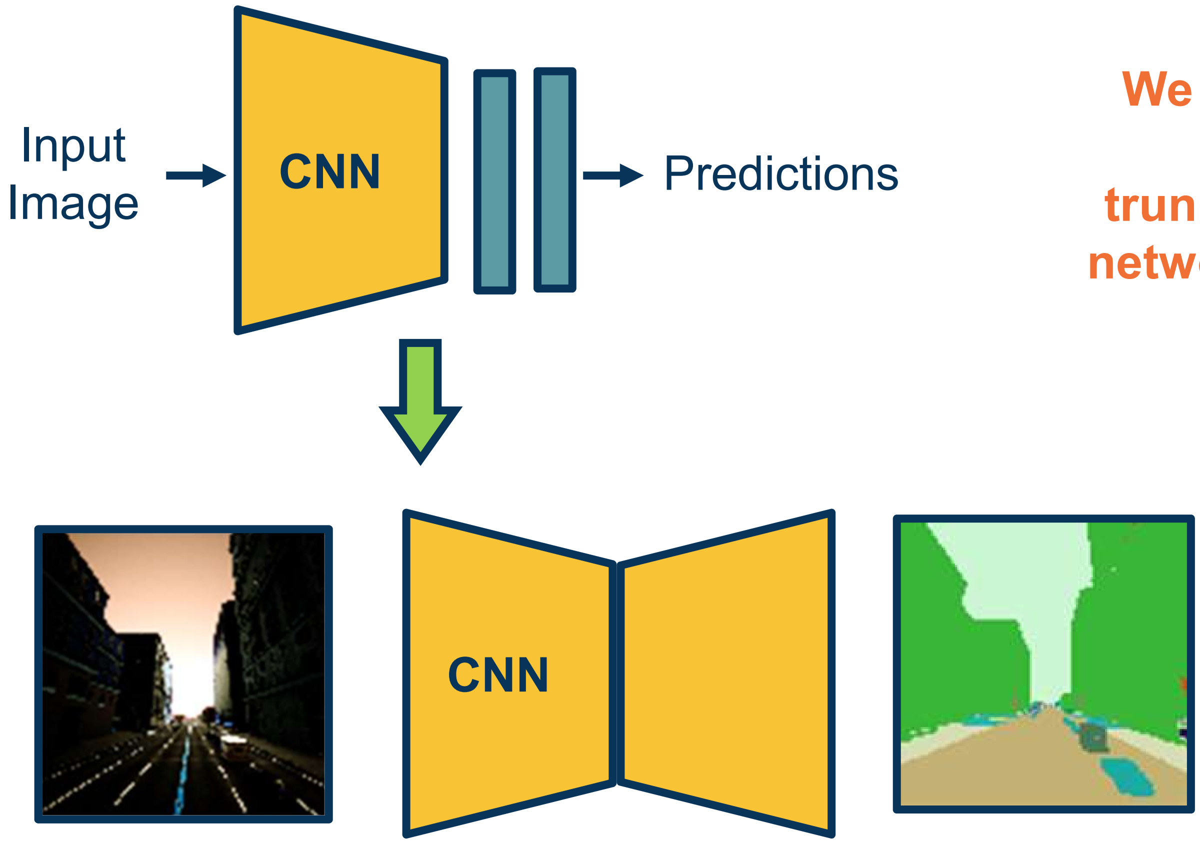
Transposed Convolution Example



We can either learn the kernels, or take corresponding encoder kernel and rotate 180 degrees (no decoder learning)

Symmetry in Encoder/Decoder

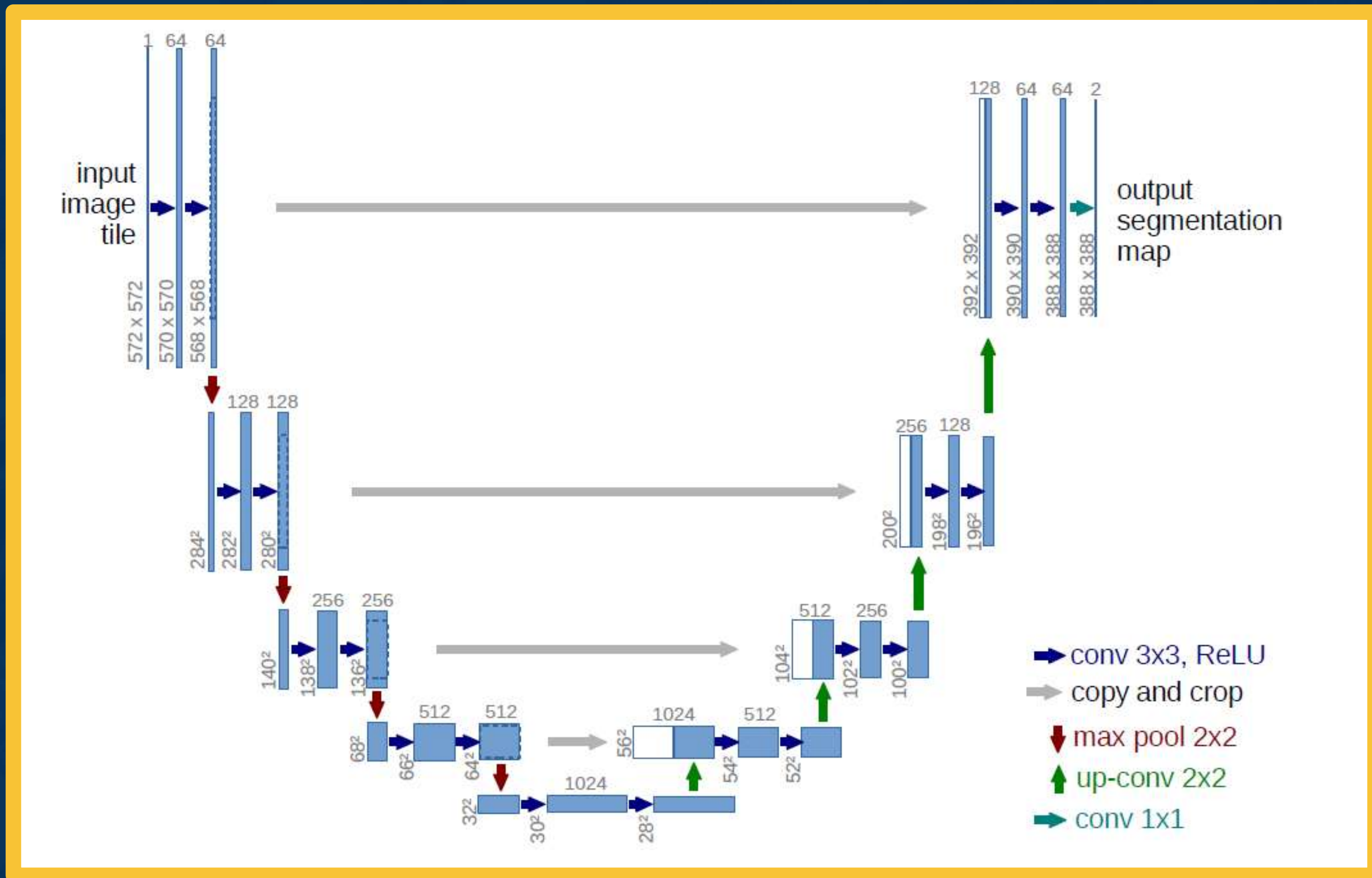
Georgia Tech



We can start with a pre-trained trunk/backbone (e.g. network pretrained on ImageNet)!

Transfer Learning

U-Net



You can have skip connections to bypass bottleneck!

Ronneberger, et al., "U-Net: Convolutional Networks for Biomedical Image Segmentation", 2015

Summary

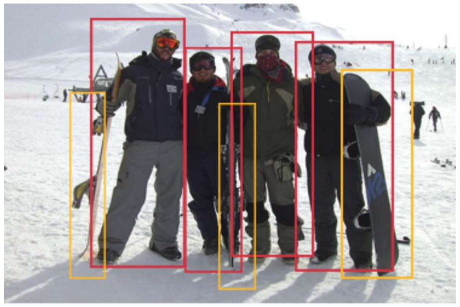
- Various ways to get **image-like outputs,** for example to predict segmentations of input images
- Fully convolutional layers essentially apply the striding idea to the output classifiers, supporting arbitrary input sizes
  - (without output size depending on what the input size is)
- We can have various upsampling layers that actually increase the size
- Encoder/decoder architectures are popular ways to leverage these to perform general image-to-image tasks

Single-Stage Object Detection

Given an image, output a list of bounding boxes with probability distribution over classes per box

Problems:

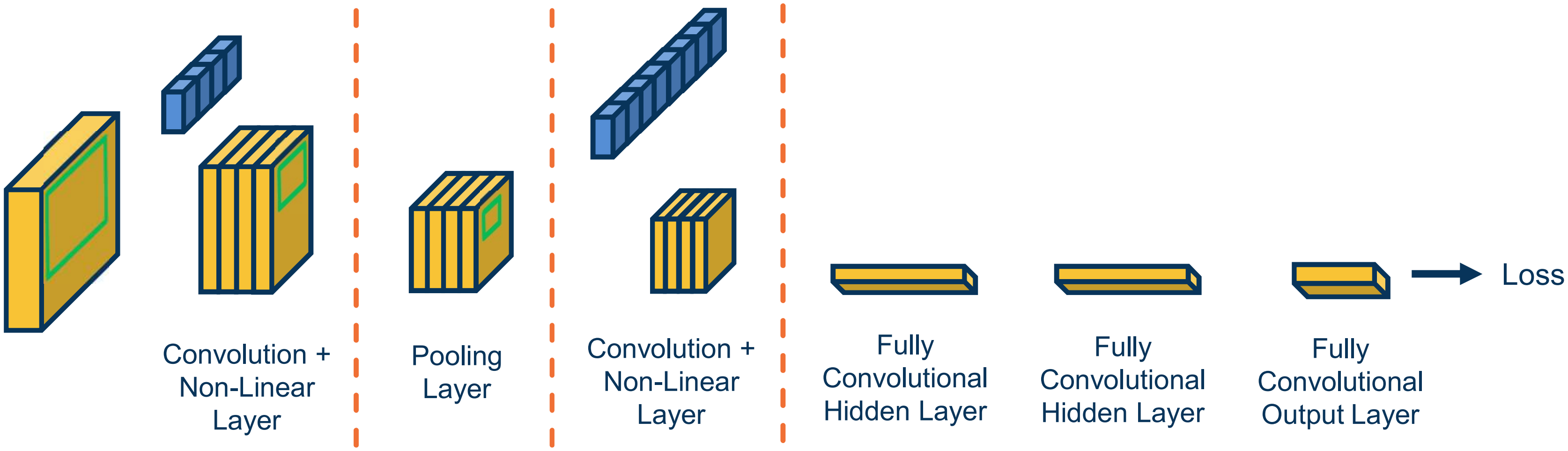
- Variable number of boxes!
- Need to determine candidate regions (position and scale) first



Object Detection

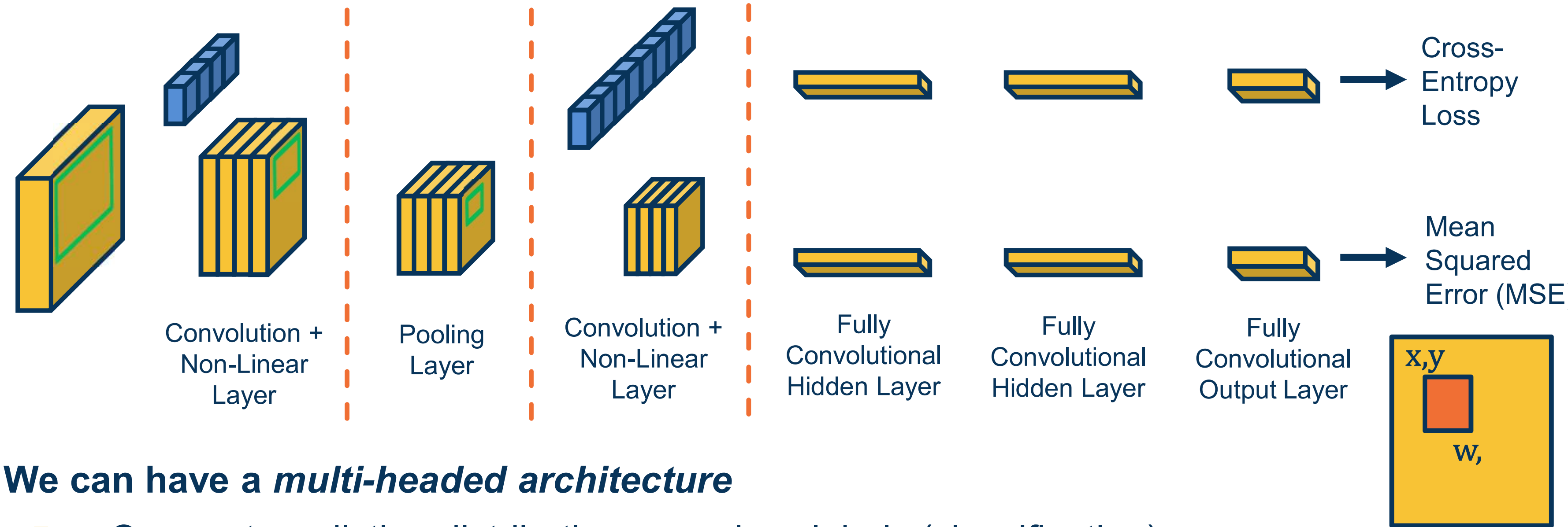
(List of bounding boxes with class distribution per box)

Object Detection Tasks



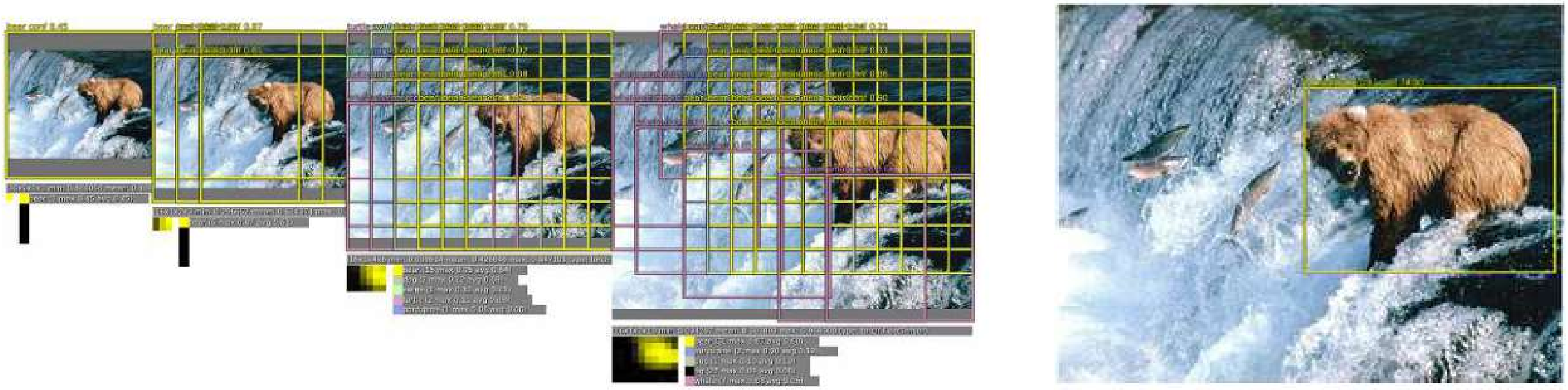
We can use the same idea of fully-convolutional networks

- Use ImageNet pre-trained model as backbone (e.g. taking in 224x224 image)
- Feed in larger image and get classifications for different windows in image



We can have a *multi-headed architecture*

- One part predicting distribution over class labels (classification)
- One part predicting a bounding box for each image region (regression)
  - Refinement to fit the object better (outputs 4 numbers)
- Both heads ***share features***! Jointly optimized (summing gradients)



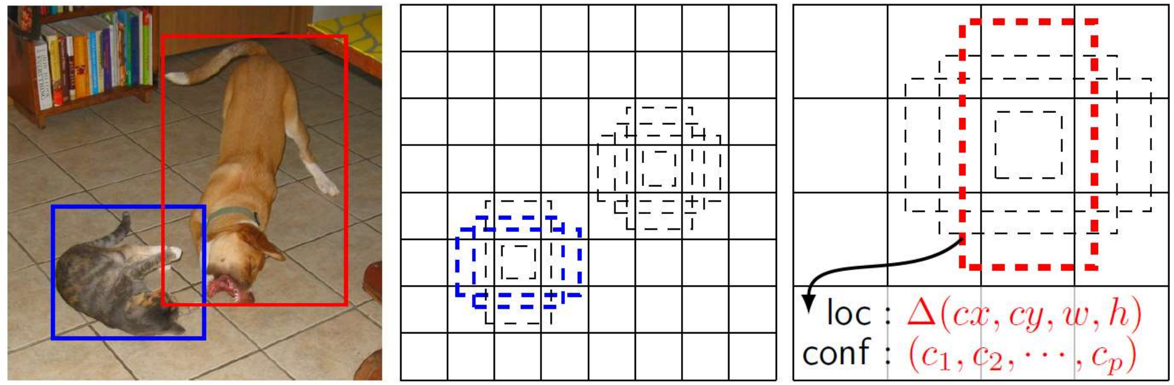
Can also do this at multiple scales to result in a large number of detections

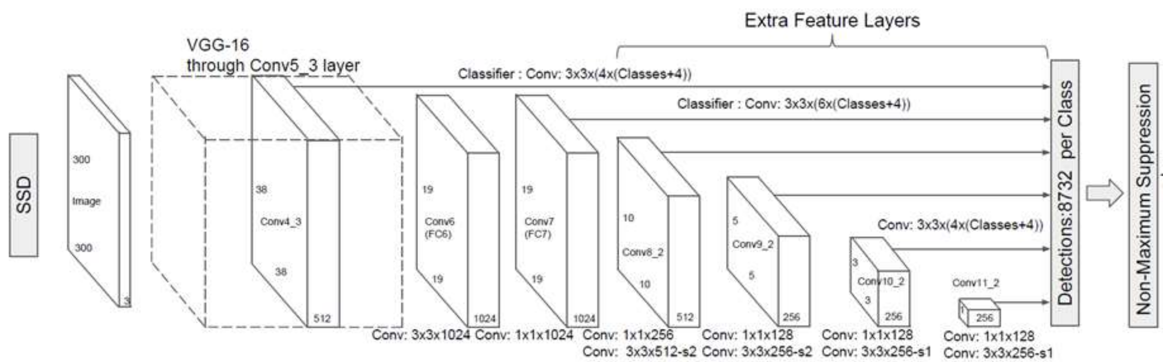
- Various tricks used to increase the resolution (decrease subsampling ratio)
- Redundant boxes are combined through **Non-Maximal Suppression (NMS)**

Sermanet, et al., "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks", 2013

Single-shot detectors use a similar idea of **grids** as anchors, with different scales and aspect ratios around them

- Various tricks used to increase the resolution (decrease subsampling ratio)





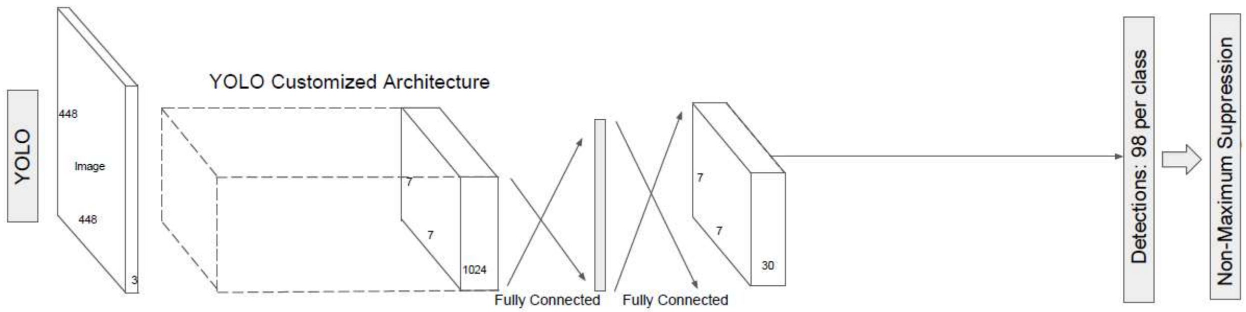
Liu, et al., "SSD: Single Shot MultiBox Detector", 2015

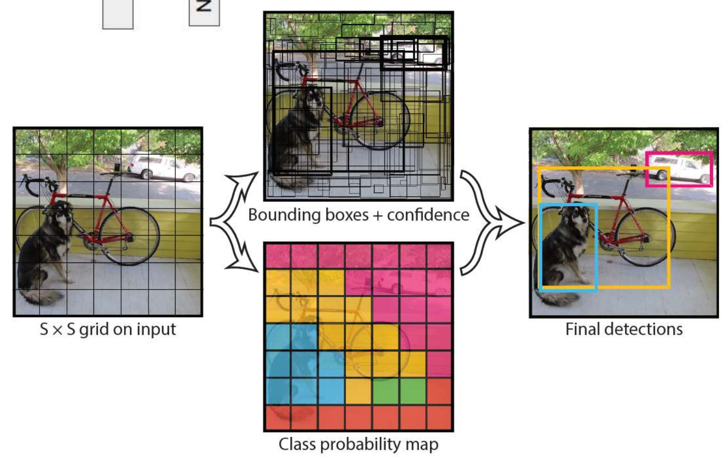
Single-Shot Detector (SSD)

Georgia Tech

You Only Look Once (YOLO)

Similar network architecture but single-scale (and hence faster for same size)



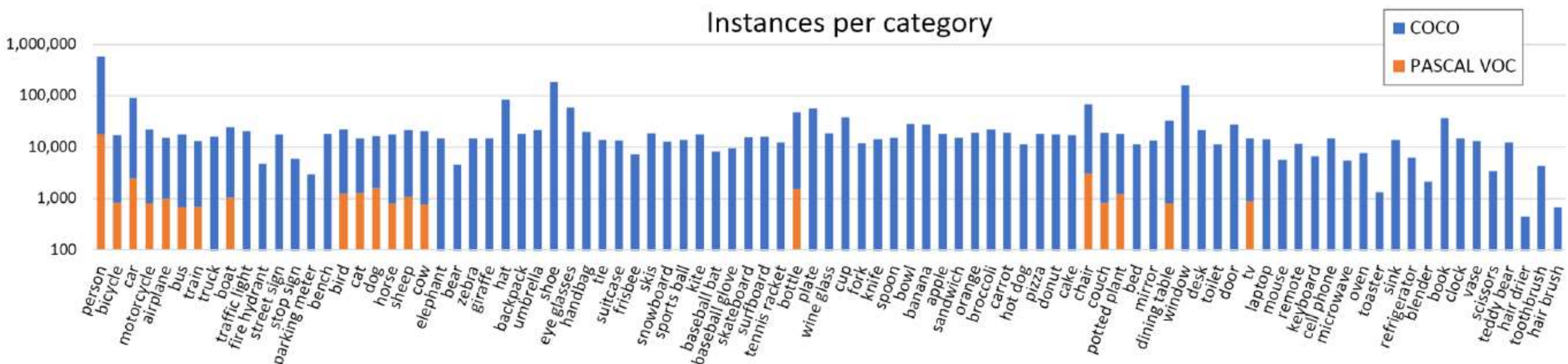


Redmon, et al., "You Only Look Once:Unified, Real-Time Object Detection", 2016

What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories



(a)

Categories per image

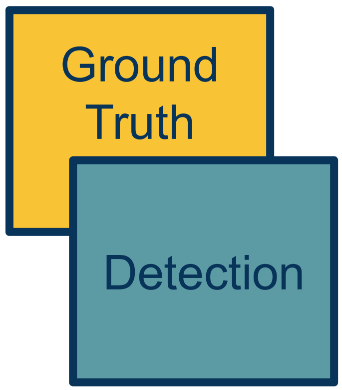
Instances per image

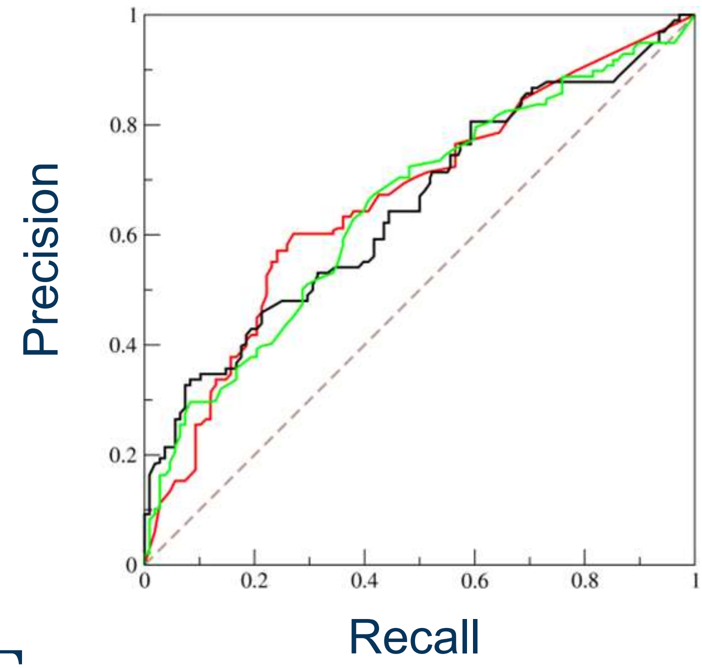
Lin, et al., "Microsoft COCO: Common Objects in Context", 2015. https://cocodataset.org/#explore

Datasets

Georgia Tech

1. For each bounding box, calculate intersection over union (IoU)
2. Keep only those with IoU > threshold (e.g. 0.5)
3. Calculate precision/recall curve across classification probability threshold
4. Calculate **average precision (AP)** over recall of [0, 0.1, 0.2, …, 1.0]
5. Average over all categories to get mean Average Precision (mAP)

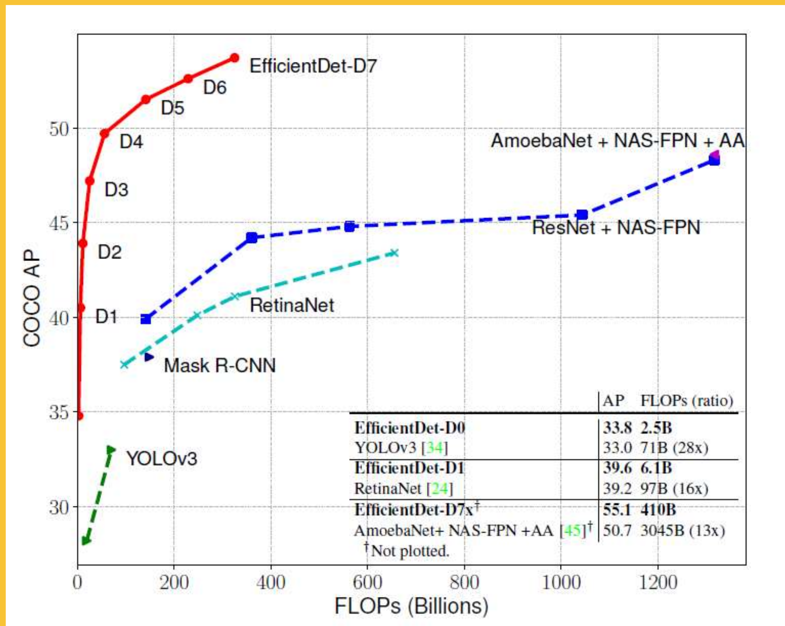


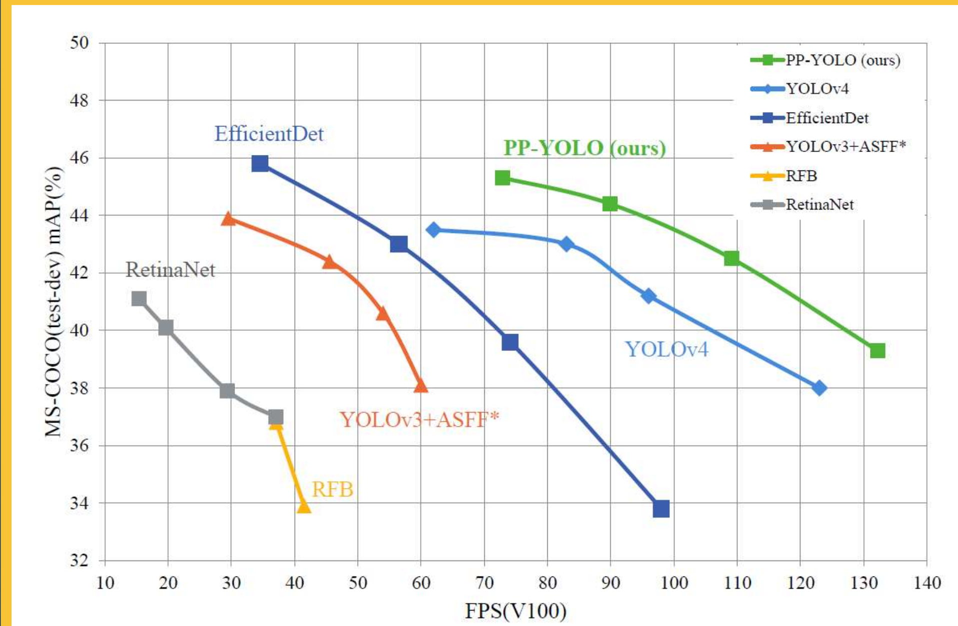


$$mAP = \frac{1}{11} \sum_{i \in [0, 0.1, \ldots 1.0]} AP_i$$

Evaluation – Mean Average Precision (mAP)

Results





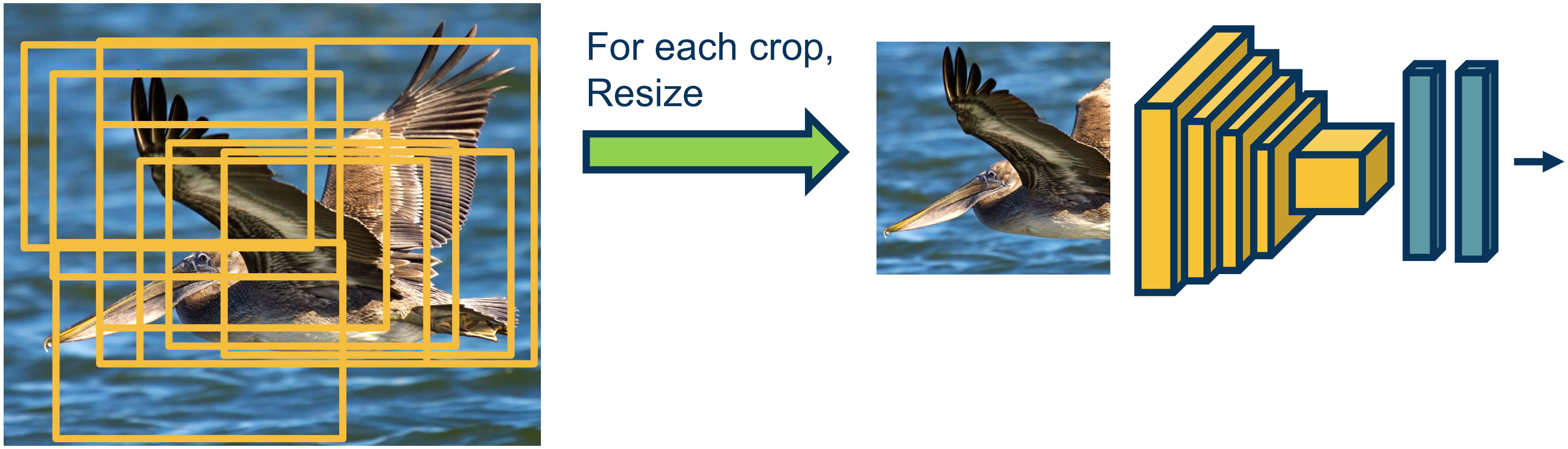
EfficientDet

PP-YOLO

Tan, et al., "EfficientDet: Scalable and Efficient Object Detection", 2020
Long et al., "PP-YOLO: An Effective and Efficient Implementation of Object Detector", 2020

Georgia Tech

Two-Stage Object Detectors



Instead of making dense predictions across an image, we can decompose the problem:

- Find regions of interest (ROIs) with object-like things
- Classifier those regions (and refine their bounding boxes)

Girshick, et al., "Rich feature hierarchies for accurate object detection and semantic segmentation", 2014

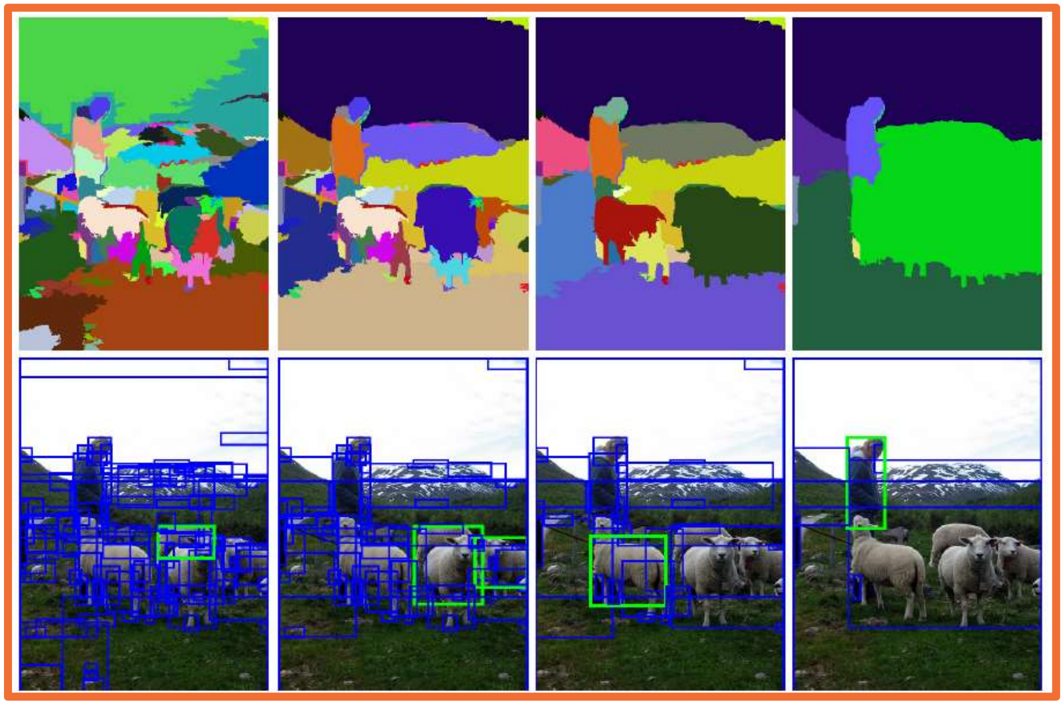
R-CNN

We can use **unsupervised (non-learned!) algorithms** for finding candidates

Downsides:

- Takes 1+ second per image
- Return thousands of (mostly background) boxes

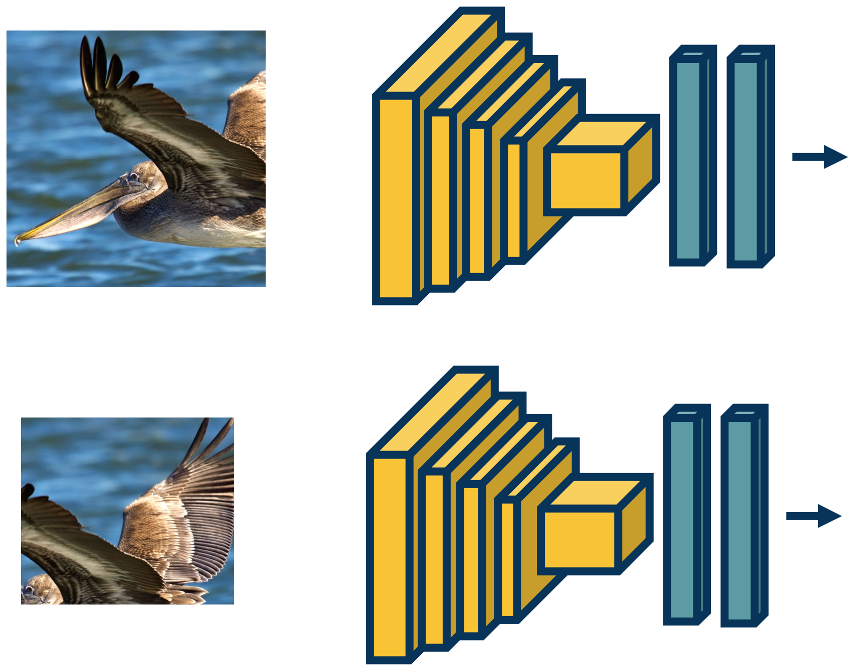
Resize each candidate to full input size and classify



Uijlings, et al., “Selective Search for Object Recognition”, 2012

Extracting Region Proposal

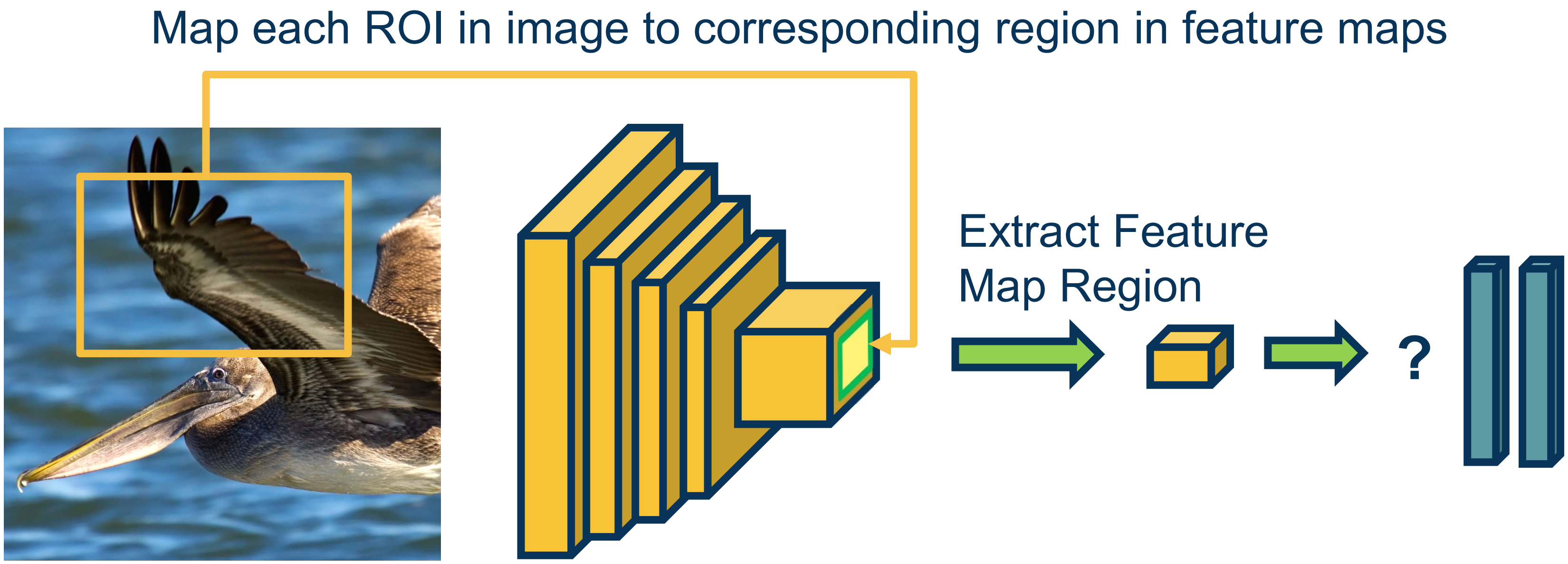
What is the problem with this?



Computation for convolutions re-done for each image patch, even if overlapping!

Girshick, et al., "Rich feature hierarchies for accurate object detection and semantic segmentation", 2014

Inefficiency of R-CNN



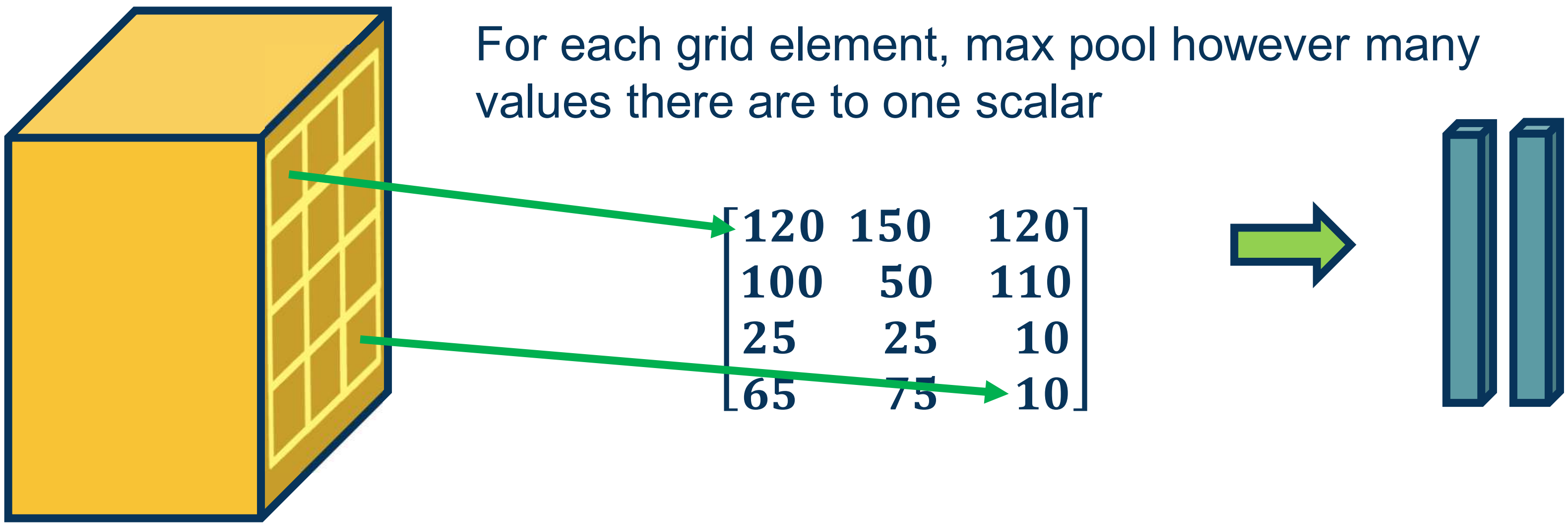
Idea: ***Reuse*** computation by finding regions in ***feature maps***

- Feature extraction only done once per image now!
- Problem: Variable input size to FC layers (different feature map sizes)

Girshick, "Fast R-CNN", 2015

Fast R-CNN

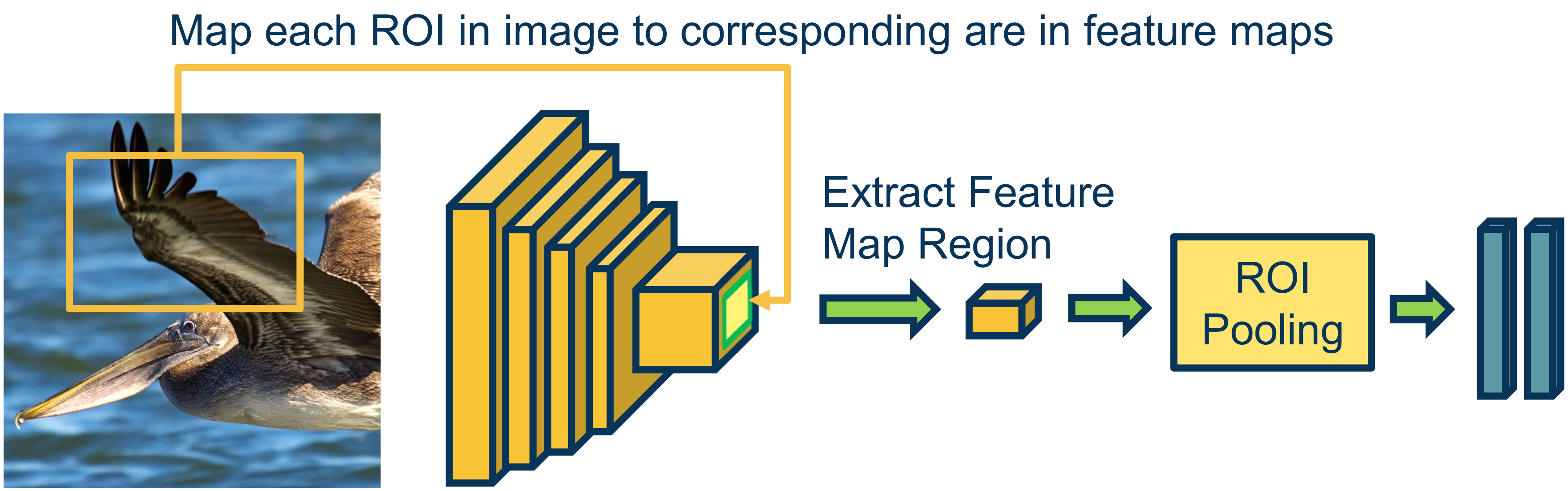
For each grid element, max pool however many values there are to one scalar



$$\begin{bmatrix} 120 & 150 & 120 \\ 100 & 50 & 110 \\ 25 & 25 & 10 \\ 65 & 75 & 10 \end{bmatrix}$$

Given an arbitrarily-sized feature map, we can use **pooling** across a grid (ROI Pooling Layer) to convert to fixed-sized representation

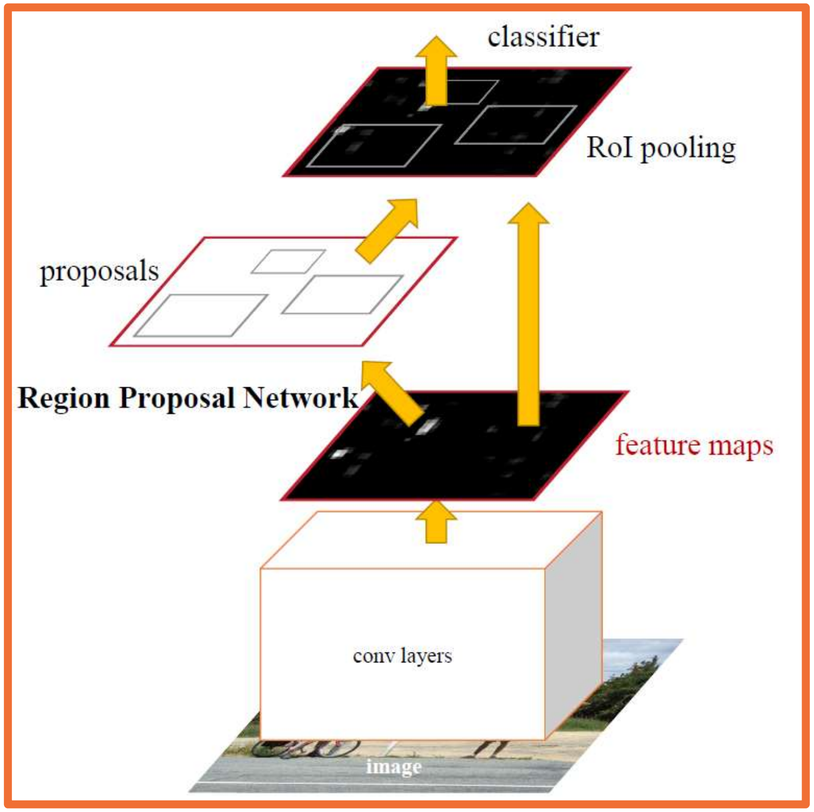
ROI Pooling



We can now train this model **end-to-end** (i.e. backpropagate through entire model including ROI Pooling)!

Fast R-CNN

- **Idea:** Why not have the neural network *also* generate the proposals?
    - Region Proposal Network (RPN) uses same features!
- Outputs ***objectness score*** and bounding box
- Top k selected for classification
- Note some parts (gradient w.r.t. bounding box coordinates) not differentiable so some complexity in implementation

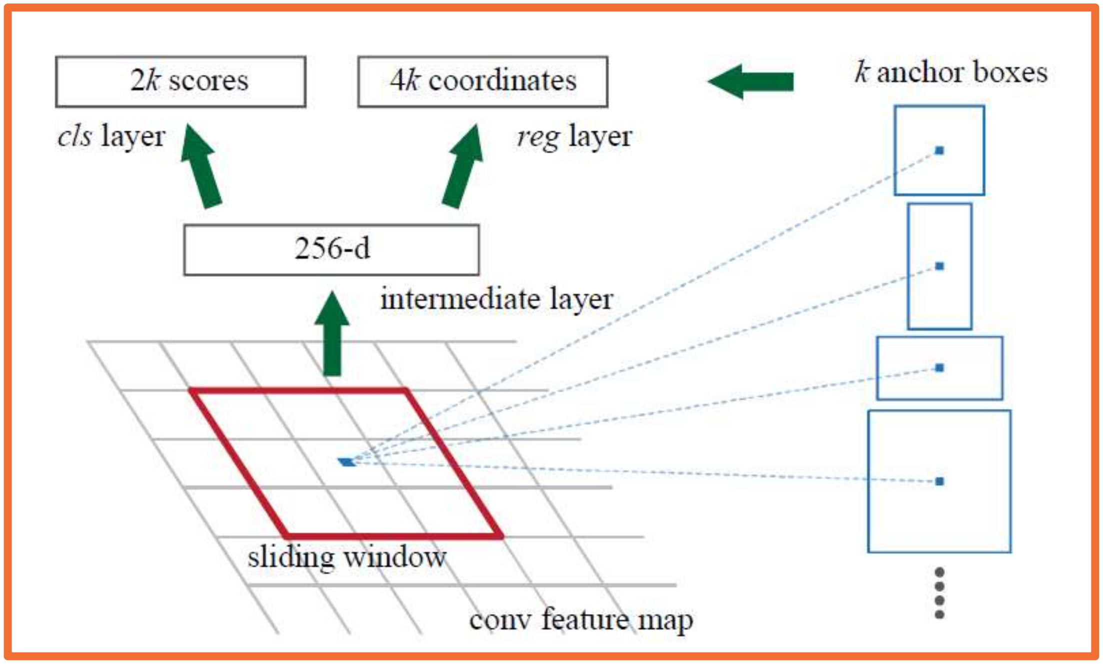


Ren, et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", 2016

Faster R-CNN

RPN also uses notion of **anchors in a grid**

Boxes of various sizes and scales classified with objectness score and refined bounding boxes refined

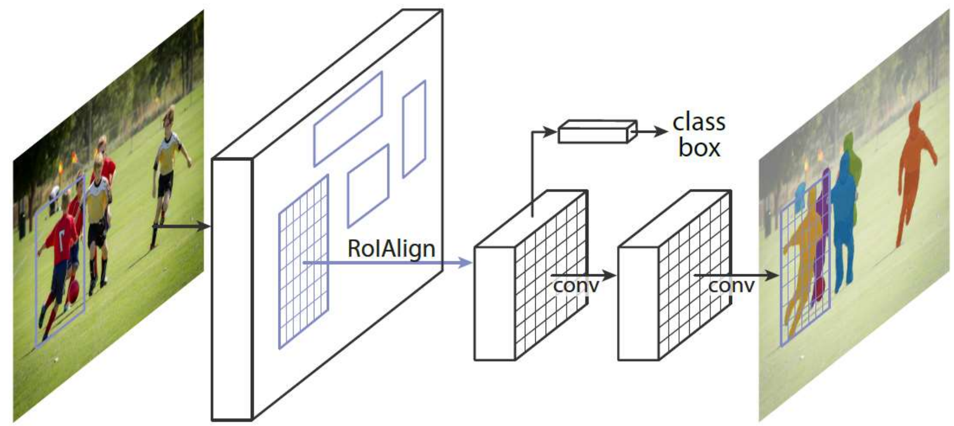


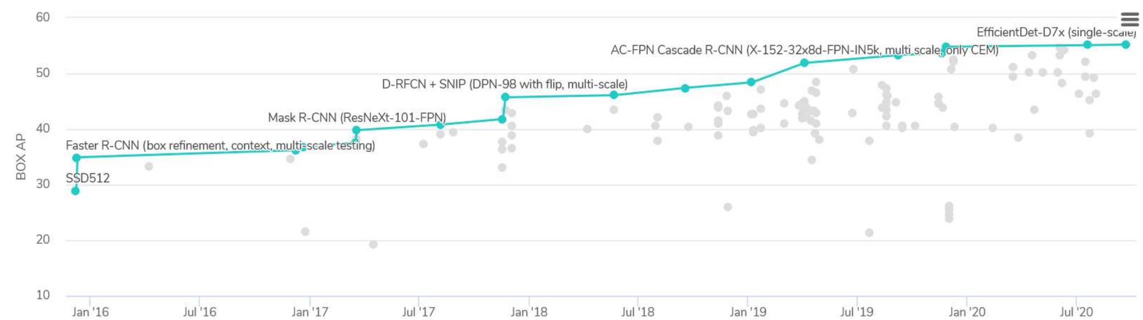
Ren, et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", 2016

Many new advancements have been made

For example, combining detection and segmentation

- Extract foreground (object) mask per bounding box





He, et al., "Mask R-CNN", 2018

https://paperswithcode.com/sota/object-detection-on-coco

Mask R-CNN

Bias & Fairness

ML and Fairness

- AI effects our lives in many ways
- Widespread algorithms with many small interactions
  - e.g. search, recommendations, social media
- Specialized algorithms with fewer but higher-stakes interactions
  - e.g. medicine, criminal justice, finance
- At this level of impact, algorithms can have unintended consequences
- Low classification error is not enough, need fairness

BUSINESS NEWS OCTOBER 10, 2018 / 3:12 AM / 6 MONTHS AGO

Amazon scraps secret AI recruiting tool that showed bias against women

Jeffrey Dastin 8 MIN READ

SAN FRANCISCO (Reuters) - Amazon.com Inc's (AMZN.O) machine-learning specialists uncovered a big problem: their new recruiting engine did not like women.

The team had been building computer programs since 2014 to review job applicants' resumes with the aim of mechanizing the search for top talent, five people familiar with the effort told Reuters.

Automation has been key to Amazon's e-commerce dominance, be it inside warehouses or driving pricing decisions. The company's experimental hiring tool used artificial intelligence to give job candidates scores ranging from one to five stars - much like

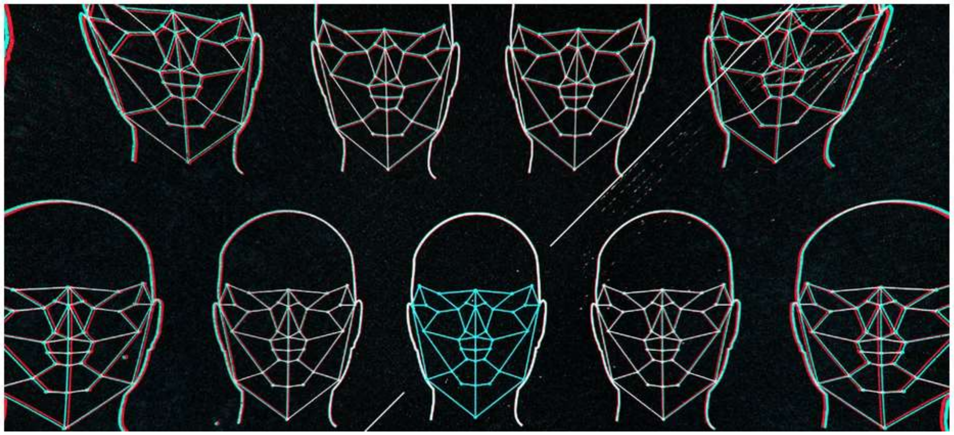
Gender and racial bias found in Amazon's facial recognition technology (again)

17

Research shows that Amazon's tech has a harder time identifying gender in darker-skinned and female faces

By James Vincent | Jan 25, 2019, 9:45am EST

SHARE



MOST READ

My Samsung Galaxy Fold screen broke after just a day

We finally know why the Instagram founders really quit

Command Line

Command Line delivers daily updates from the near-future.

Chester the AI Radiology Assistant

NOT FOR MEDICAL USE. This is a web based (but locally run) prototype system for diagnosing chest X-ray images. The patient data remains on your computer and all computation occurs in your browser. The goals of this system are:

1. Let people play with deep learning tools to know how they work and their limitations.
2. Show the potential of open data (needed to build a public system like this).
3. Create a tool to help teach radiology.
4. Demonstrate a model delivery system that can scale to provide free medical tools to the world.

Why is this a problem?

Decision Theory

Define a loss function L(y,a)

y

a

	None	Lung	Breast
Surgery	100	20	10
No surgery	0	50	50

Pick the action with minimum expected loss (risk)

$$a^*(x) = \arg\min_a \sum p(y|x) L(y, a)$$

Equal costs -> Cross Entropy!

ML and Fairness

- Fairness is morally and legally motivated
- Takes many forms
- Criminal justice: recidivism algorithms (COMPAS)
  - Predicting if a defendant should receive bail
  - Unbalanced false positive rates: more likely to wrongly deny a black person bail

Table 1: ProPublica Analysis of COMPAS Algorithm

	White	Black
Wrongly Labeled High-Risk	23.5%	44.9%
Wrongly Labeled Low-Risk	47.7%	28.0%

https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

Example – Word Embeddings

- Fairness is morally and legally motivated
- Takes many forms
- Bias found in word embeddings (Bolukbasi et al. 2016)
  - Examined word embeddings (word2vec) trained on Google News
  - Represent each word with high-dimensional vector
  - Vector arithmetic: analogies like Paris - France = London - England
  - Found also: man - woman = programmer - homemaker = surgeon - nurse
- The good news: word embeddings learn so well!
- The bad news: sometimes too well
- Our chatbots should be less biased than we are

Example – Word Embeddings

- **Algorithmic fairness**: how can we ensure that our algorithms act in ways that are fair?
  - This definition is vague and somewhat circular
  - Describes a broad set of problems, not a specic technical approach
  - Related to **accountability**: who is responsible for automated behaviour? How do we supervise/audit machines which have large impact?
  - Also **transparency**: why does an algorithm behave in a certain way? Can we understand its decisions? Can it explain itself?
  - Connections to **AI safety** and **aligned AI**: how can we make AI without unintended negative consequences? Aligns with our values?

Why Fairness is Hard

- Suppose we are a bank trying to fairly decide who should get a loan
  - i.e. Who is most likely to pay us back?
- Suppose we have two groups, A and B (the sensitive attribute)
  - This is where discrimination could occur
- The simplest approach is to remove the sensitive attribute from the data, so that our classier doesn't know the sensi

Table 2: To Loan or Not to Loan?

Age	Gender	Postal Code	Req Amt	A or B?	Pay
46	F	M5E	\$300	A	1
24	M	M4C	\$1000	B	1
33	M	M3H	\$250	A	1
34	F	M9C	\$2000	A	0
71	F	M3B	\$200	A	0
28	M	M5W	\$1500	B	0

Why Fairness is Hard

- However, if the sensitive attribute is correlated with the other attributes, this isn't good enough
- It is easy to predict race if you have lots of other information (e.g. home address, spending patterns)
- More advanced approaches are necessary

Table 3: To Loan or Not to Loan? (masked)

Age	Gender	Postal Code	Req Amt	A or B?	Pay
46	F	M5E	$300	?	1
24	M	M4C	$1000	?	1
33	M	M3H	$250	?	1
34	F	M9C	$2000	?	0
71	F	M3B	$200	?	0
28	M	M5W	$1500	?	0

Definitions of Fairness – Group Fairness

- So we've built our classier . . . how do we know if we're being fair?
- One metric is demographic parity | requiring that the same percentage of A and B receive loans
  - What if 80% of A is likely to repay, but only 60% of B is?
  - Then demographic parity is too strong
- Could require equal false positive/negative rates
  - When we make an error, the direction of that error is equally likely for both groups

$$P(loan|no\ repay, A) = P(loan|no\ repay, B)$$
$$P(no\ loan|would\ repay, A) = P(no\ loan|would\ repay, B)$$

- These are definitions of group fairness
- Treat different groups equally"

Definitions of Fairness – Individual Fairness

- Also can talk about individual fairness | “Treat similar examples similarly"
- Learn fair representations
  - Useful for classification, not for (unfair) discrimination
  - Related to domain adaptation
  - Generative modelling/adversarial approaches

(a) Unfair representations (b) Fair(er) representations

Figure 1: “The Variational Fair Autoencoder” (Louizos et al., 2016)

Conclusion

- This is an exciting field, quickly developing
- Central definitions still up in the air
- AI moves fast | lots of (currently unchecked) power
- Law/policy will one day catch up with technology
- Those who work with AI should be ready
  - **Think about implications of what you develop!**