Topics:

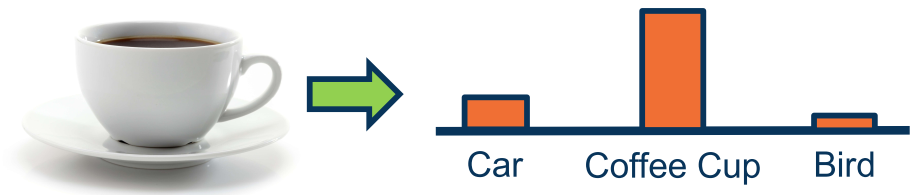
- Advanced Architectures
- Bias, Fairness, Calibration

CS 4644-DL / 7643-A
ZSOLT KIRA

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

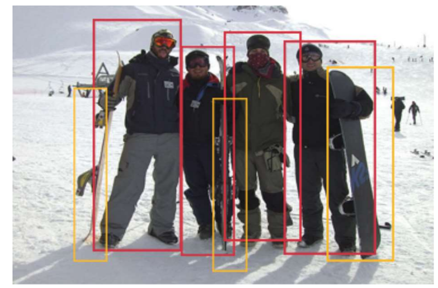
- **Projects**
  - Project proposal due **March 13th**
  - **Next class: Come with project teams/ideas and run them by TAs!**

- **Meta Office Hours on Fairness/Bias Friday 3pm EST**
  - NOT recorded!



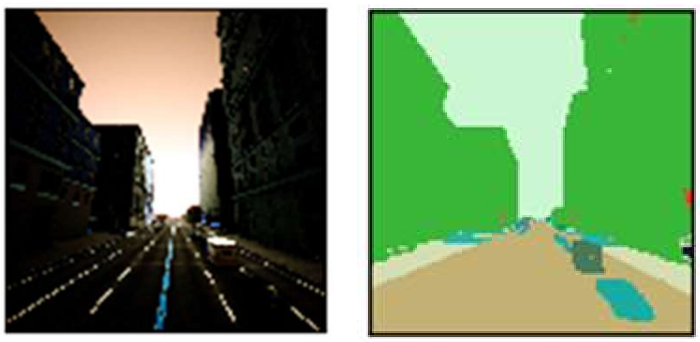
Classification

(Class distribution per image)



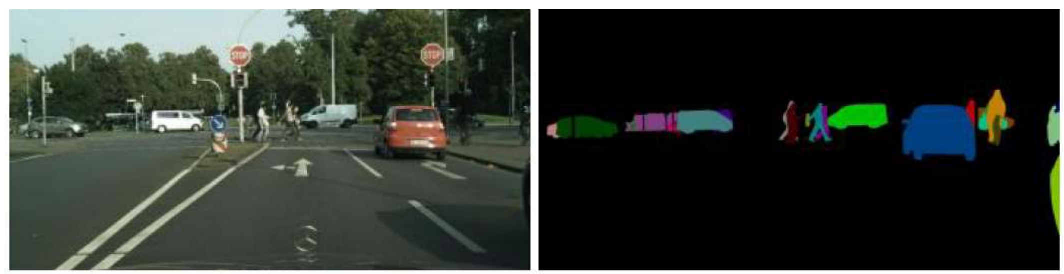
Object Detection

(List of bounding boxes with class distribution per box)



Semantic Segmentation

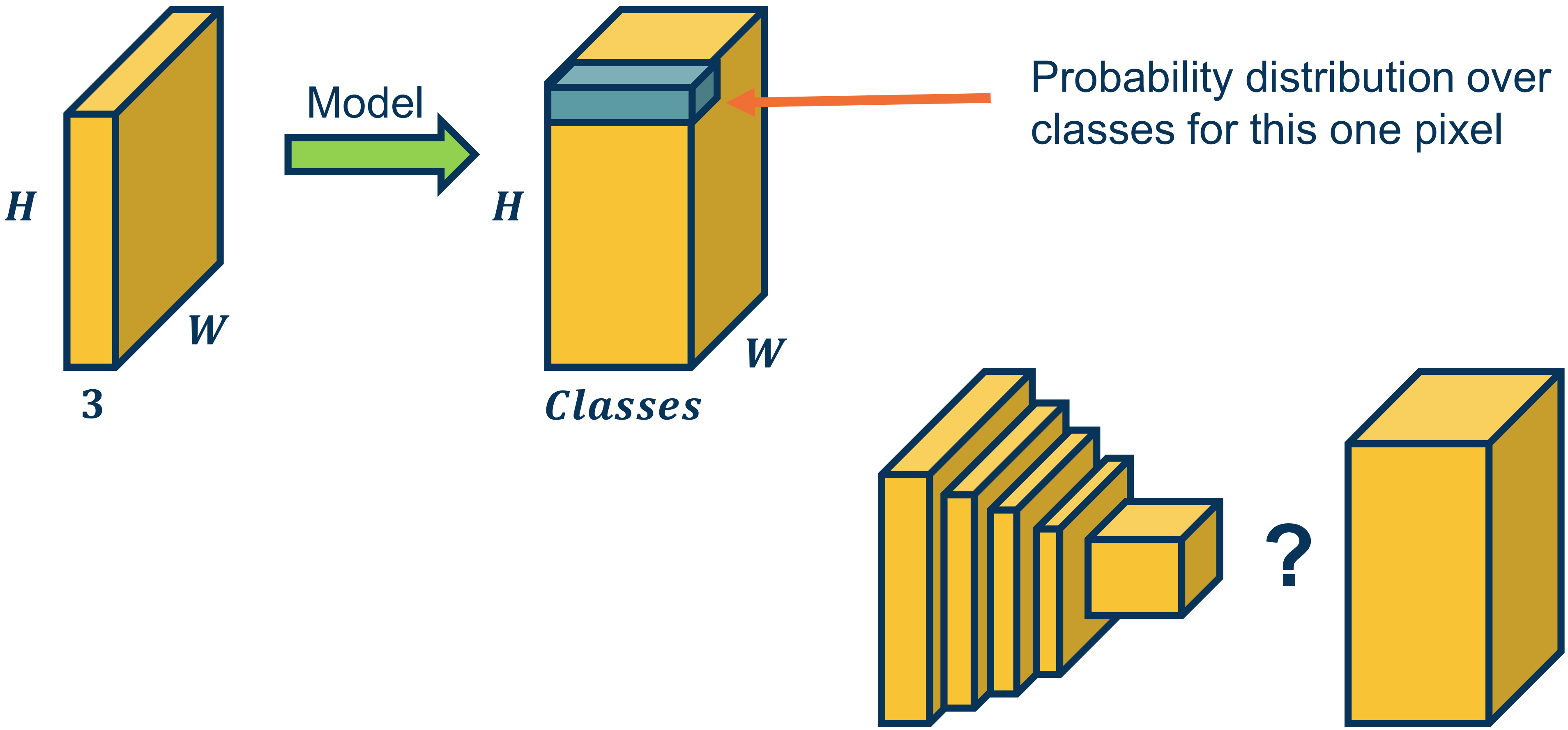
(Class distribution per pixel)



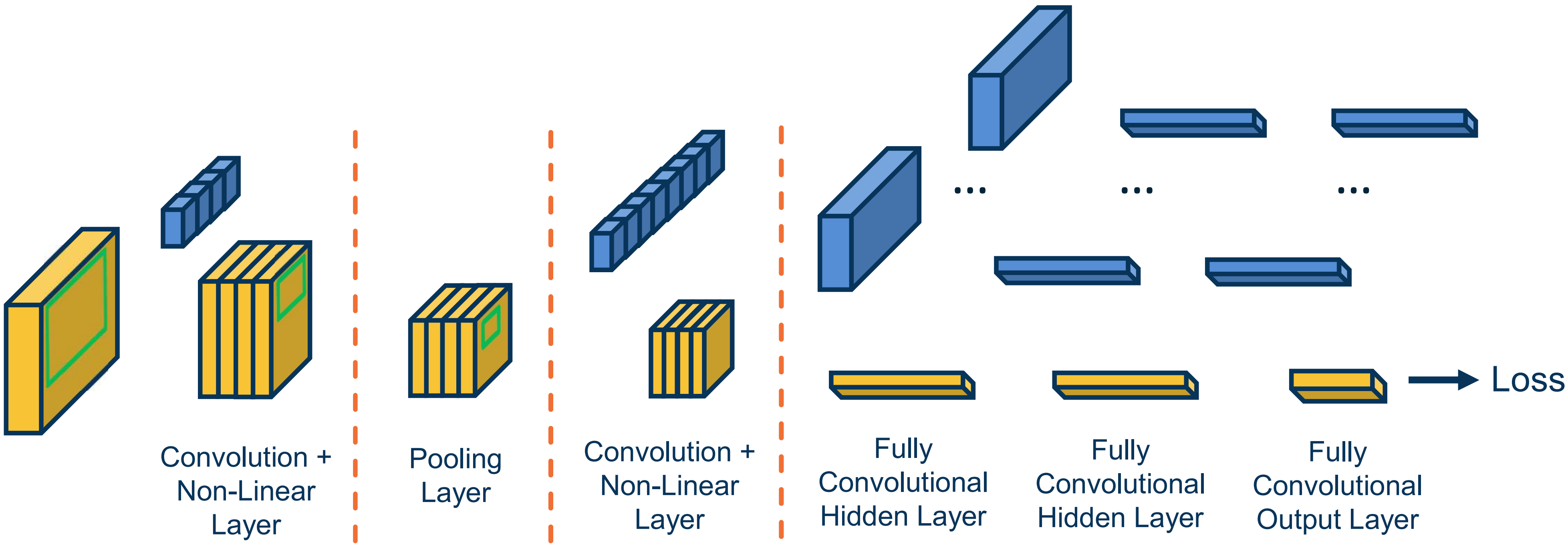
Instance Segmentation

(Class distribution per pixel with unique ID)

Computer Vision Tasks



Input & Output



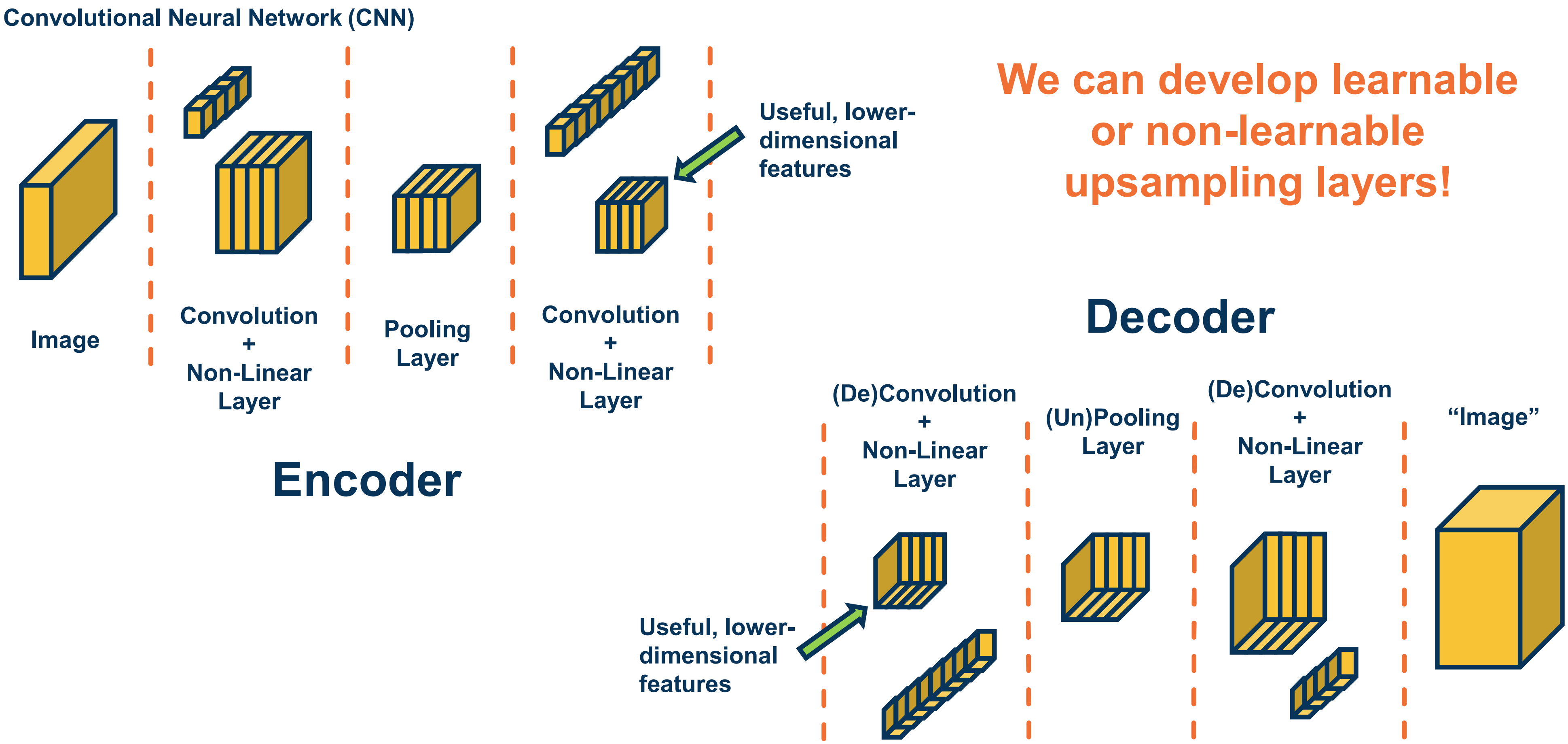
Each kernel has the size of entire input! (output is 1 scalar)

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

Converting FC Layers to Conv Layers

Idea 2: “De”Convolution and UnPooling

We can develop learnable or non-learnable upsampling layers!

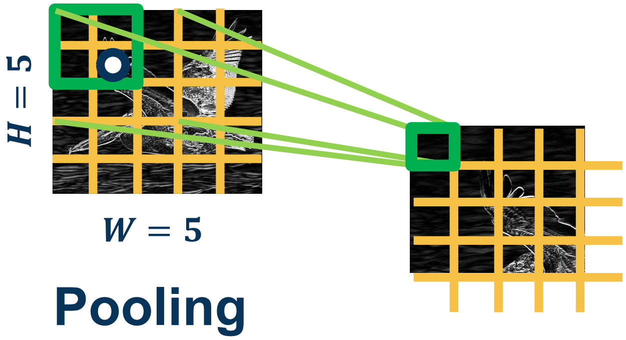


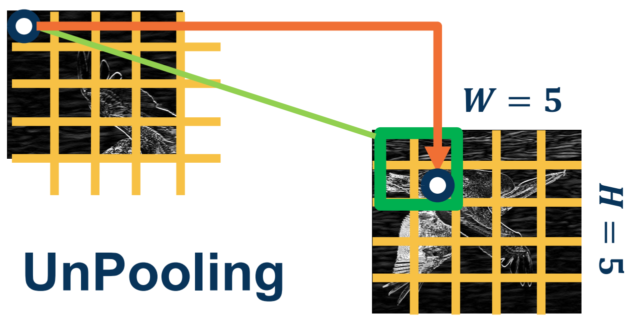
Example : Max pooling

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

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

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





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)

Georgia Tech

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

Contributions from multiple windows are summed

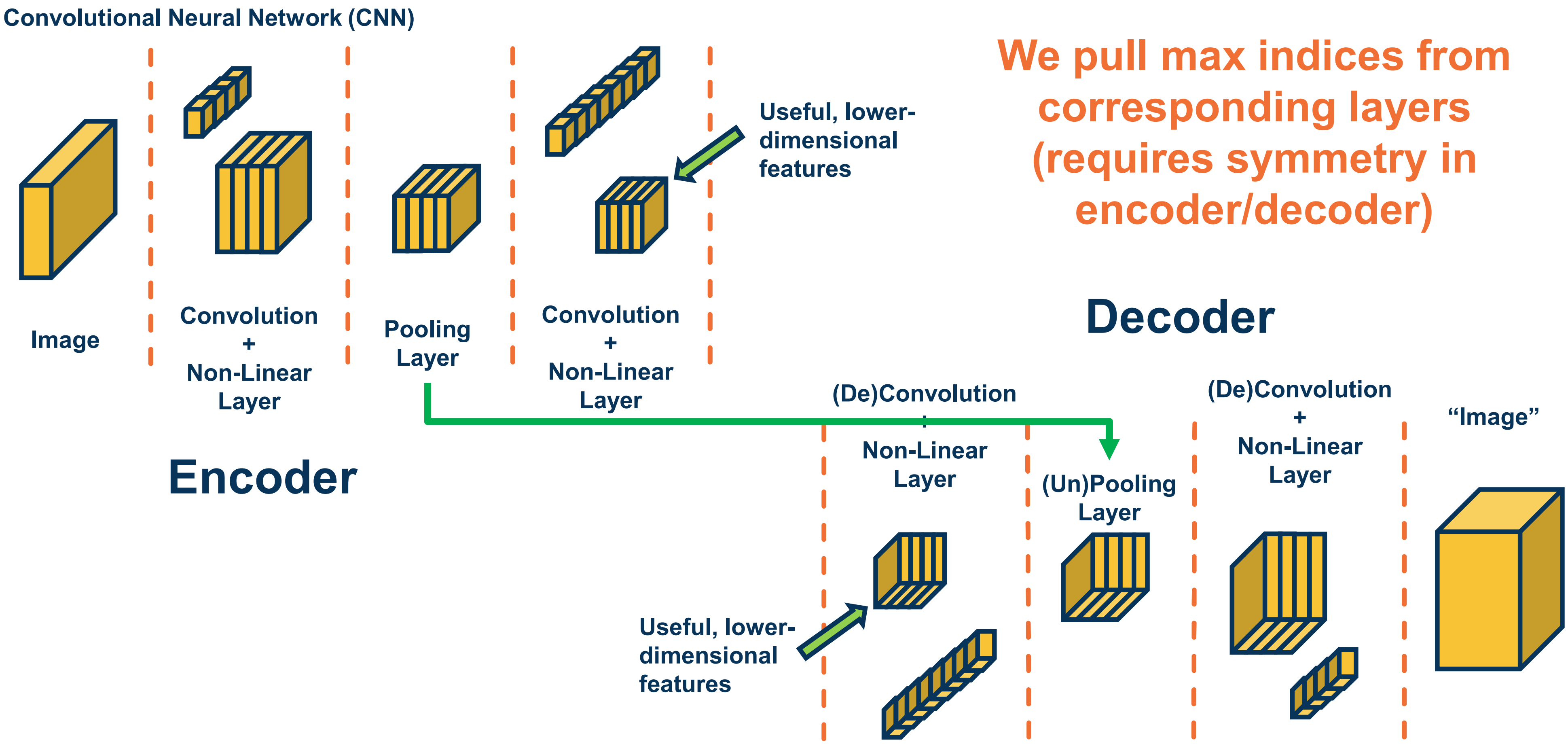
Encoder

Decoder

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

Max Unpooling Example

Georgia Tech

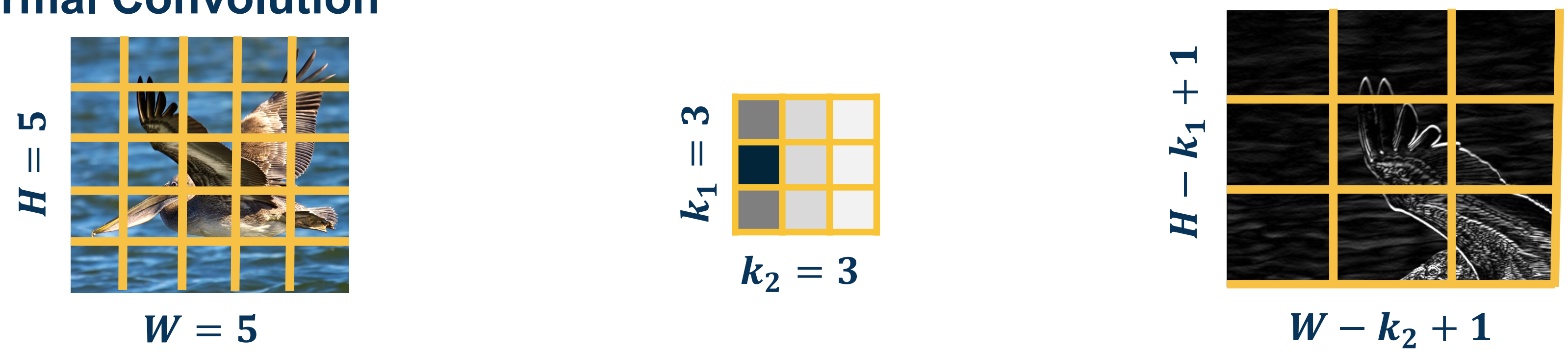


We pull max indices from corresponding layers (requires symmetry in encoder/decoder)

Symmetry in Encoder/Decoder

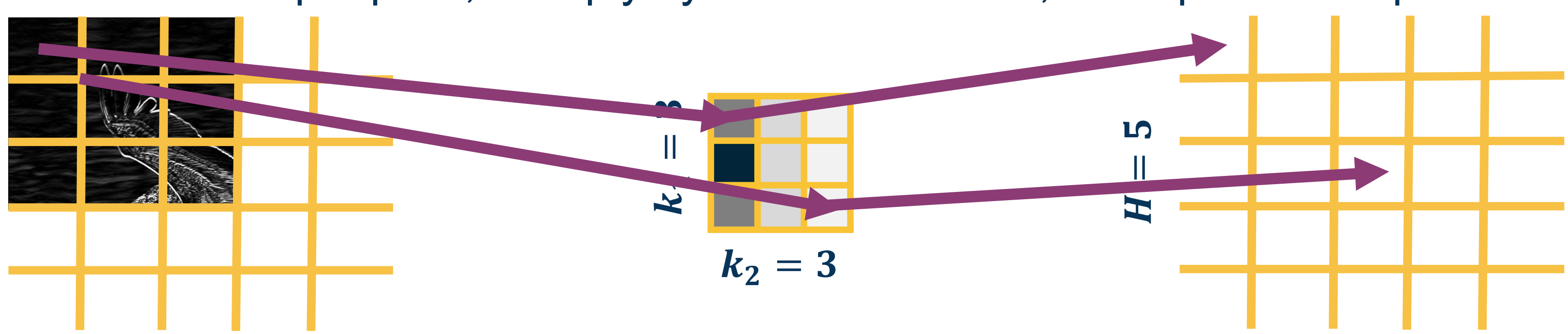
How can we *upsample* using convolutions and learnable kernel?

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)

Georgia Tech

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

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

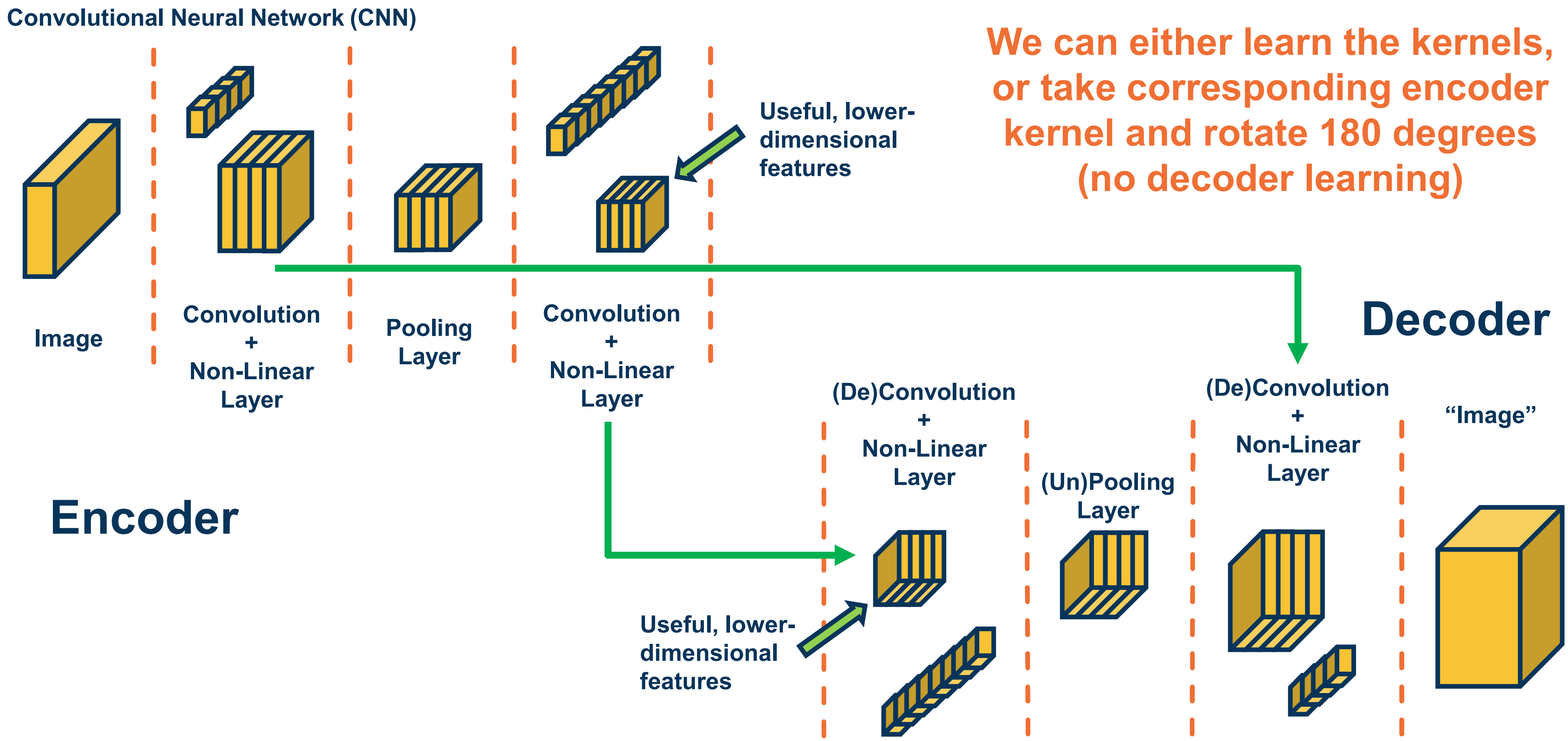
$X(0,0)$

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

Incorporate

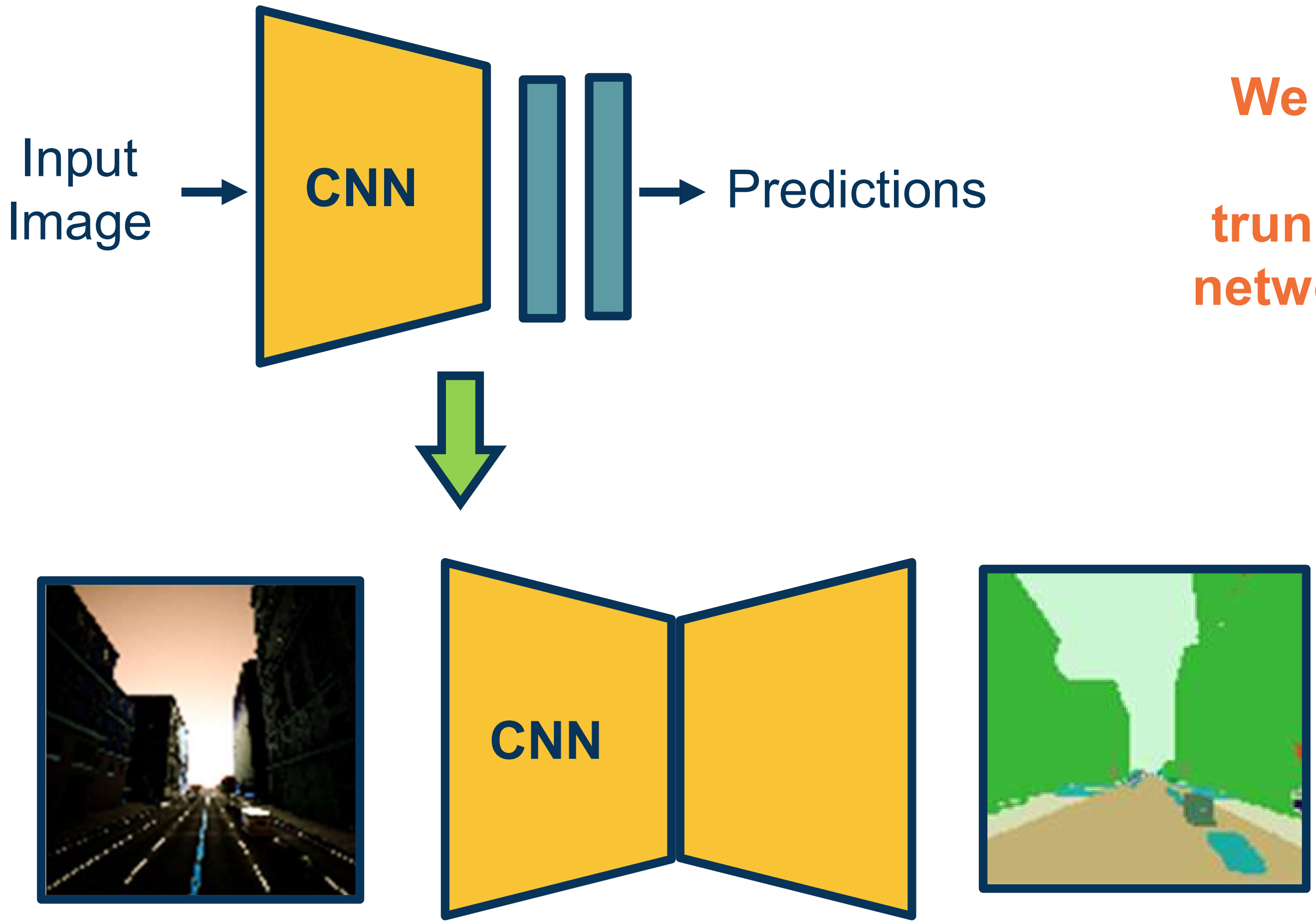
$X(1,0)$

Transposed Convolution Example



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

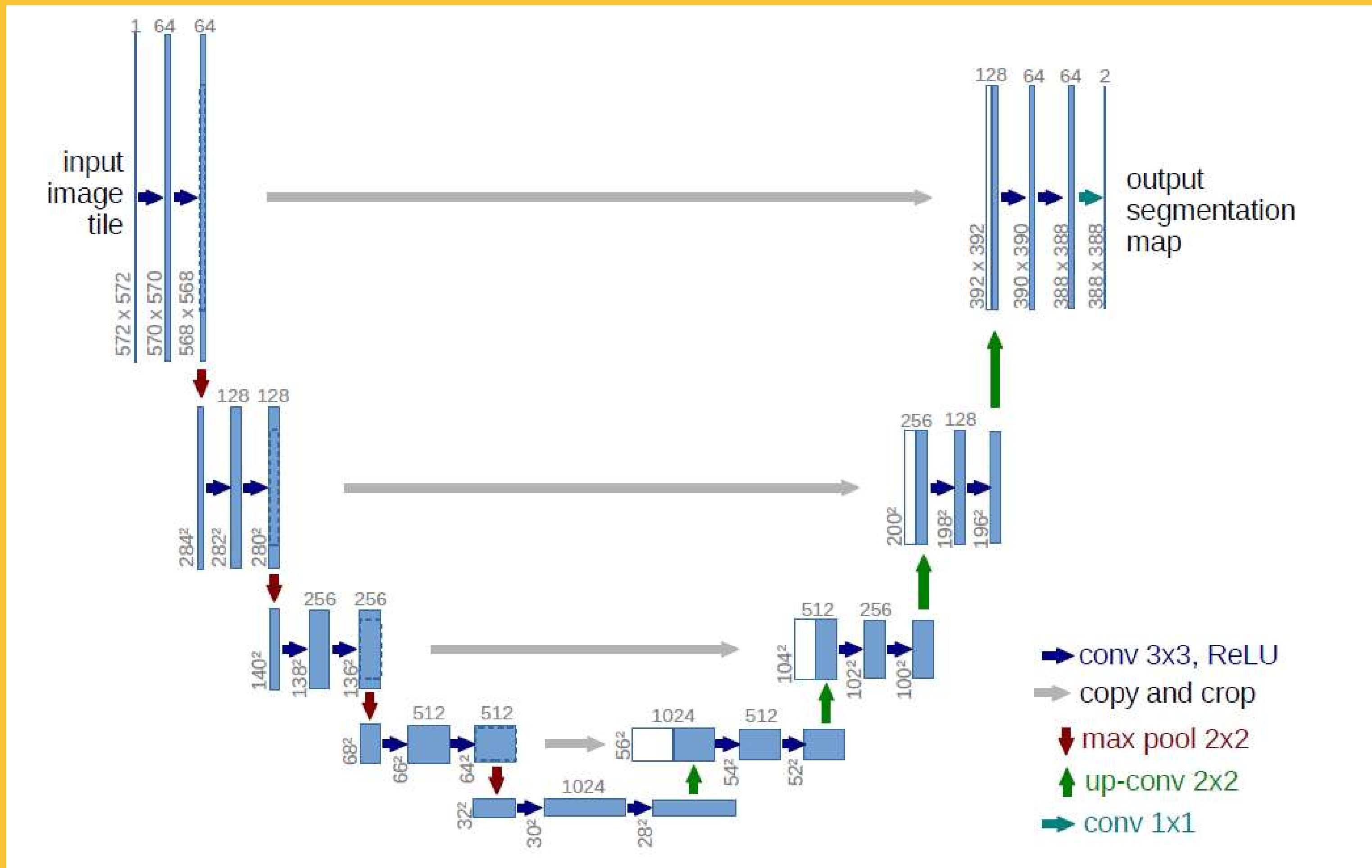
Symmetry in Encoder/Decoder



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

U-Net

You can have skip connections to bypass bottleneck!



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

Georgia Tech

Summary

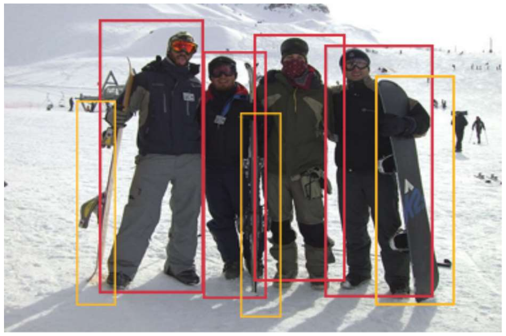
- Various ways to get **image-like outputs,** for example to predict segmentations of input images
- Fully convolutional layers essentially apply the striding idea to the output classifiers, supporting arbitrary input sizes
  - (without output size depending on what the input size is)
- We can have various 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

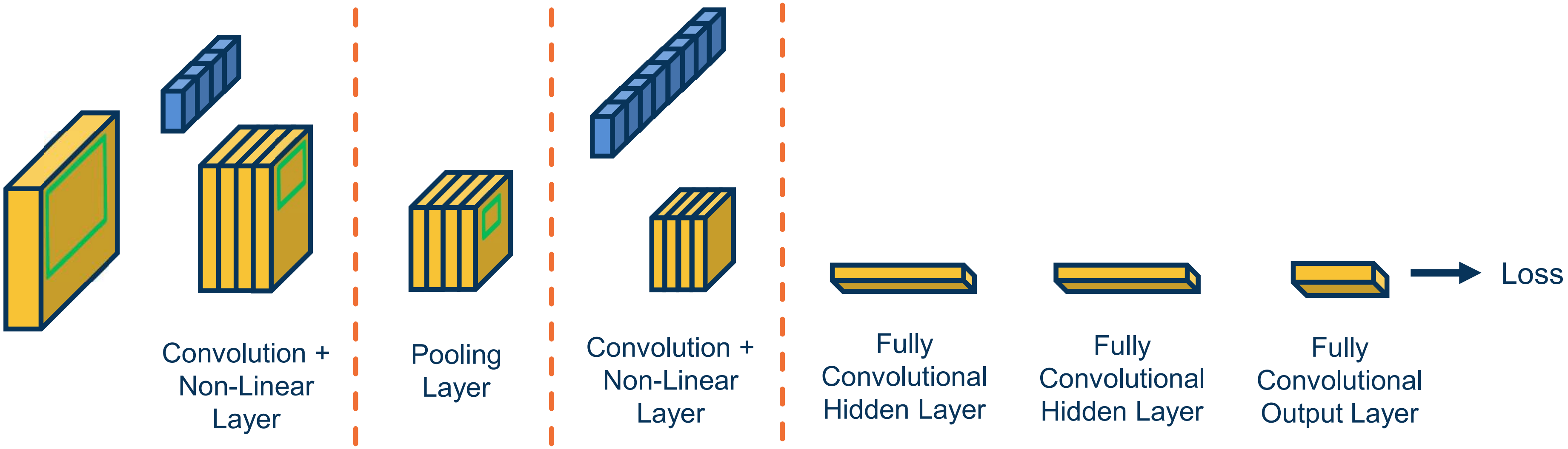
Problems:

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



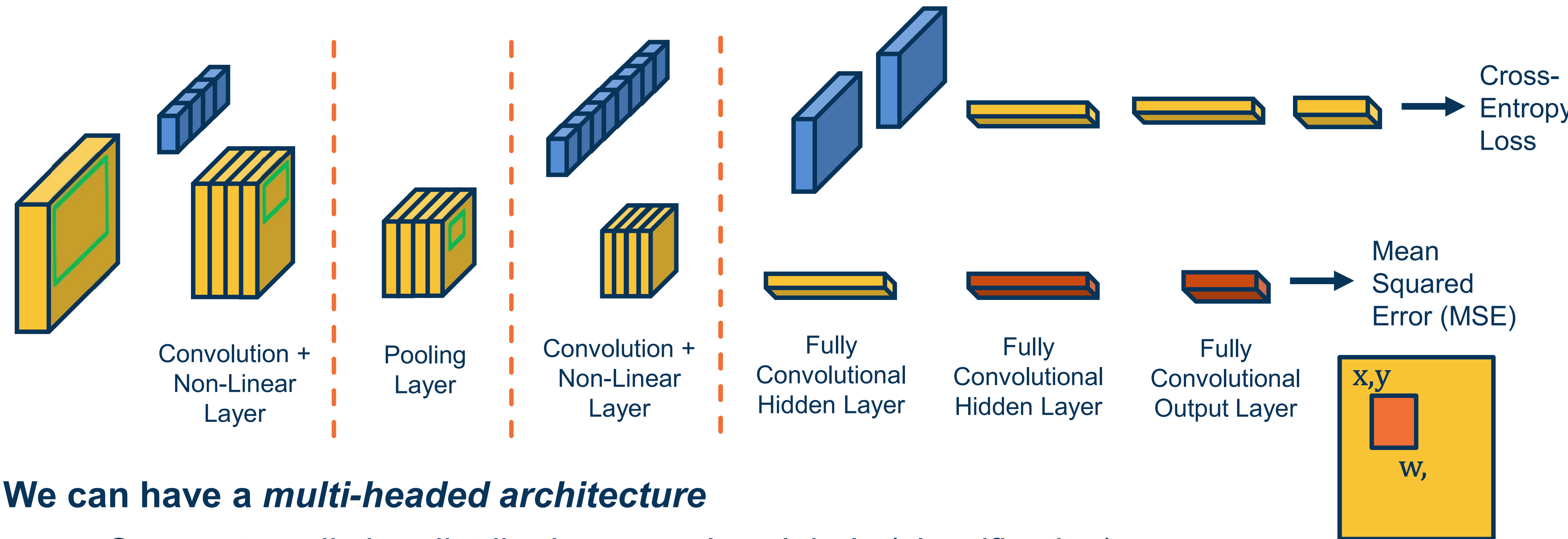
Object Detection

(List of bounding boxes with class distribution per box)



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)

Object Detection Tasks

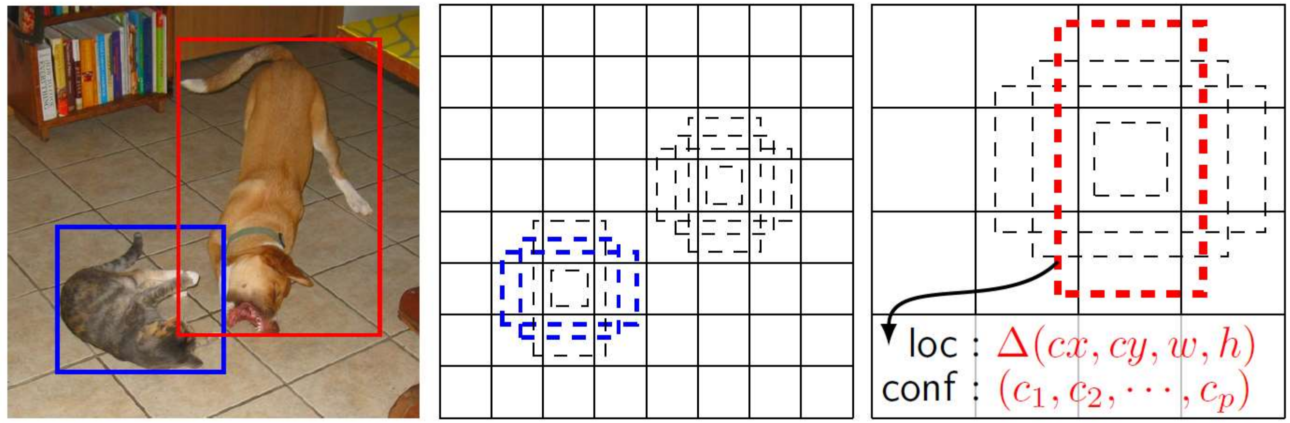
Can also do this at multiple scales to result in a large number of detections

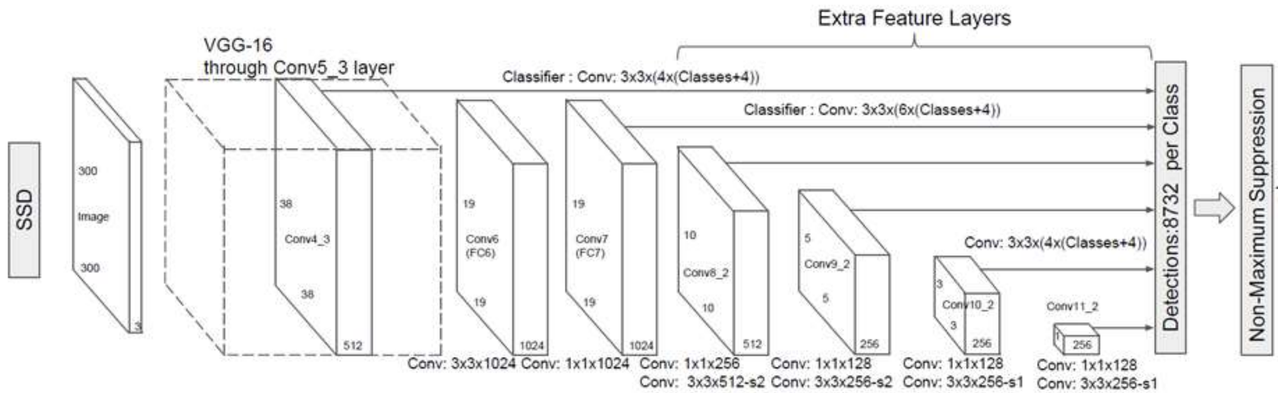
- Various tricks used to increase the resolution (decrease subsampling ratio)
- Redundant boxes are combined through **Non-Maximal Suppression (NMS)**

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)

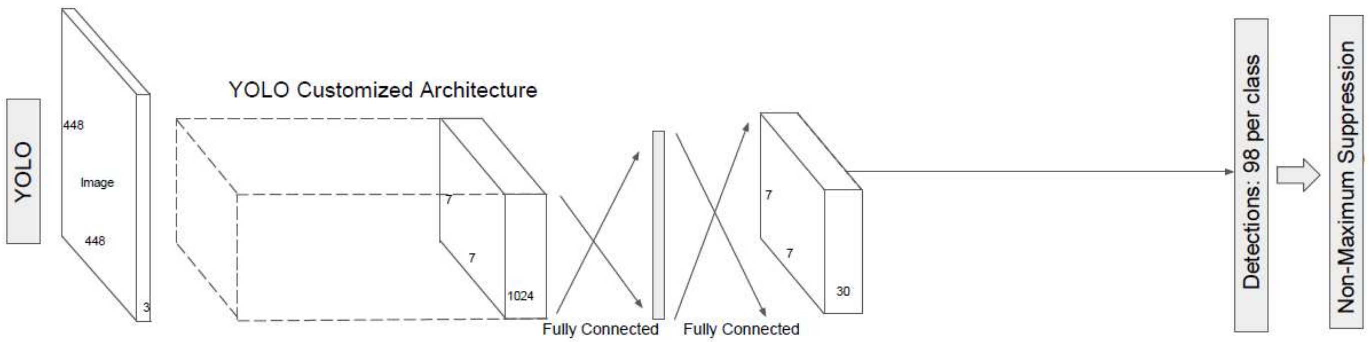


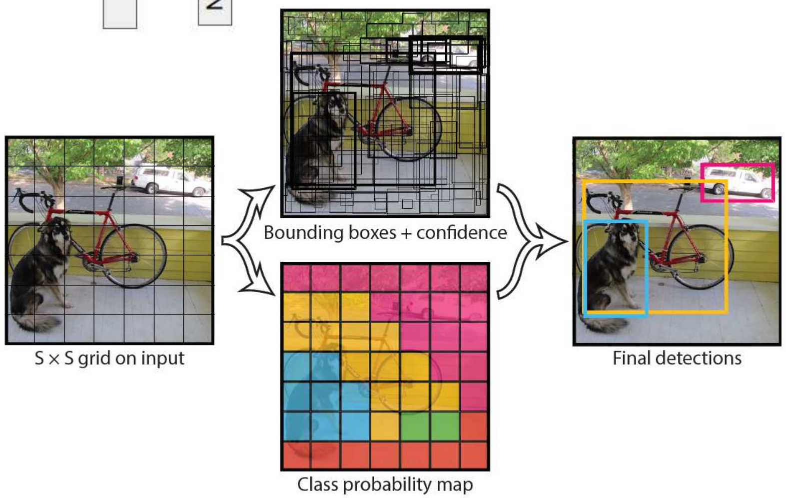


Liu, et al., "SSD: Single Shot MultiBox Detector", 2015

Single-Shot Detector (SSD)

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

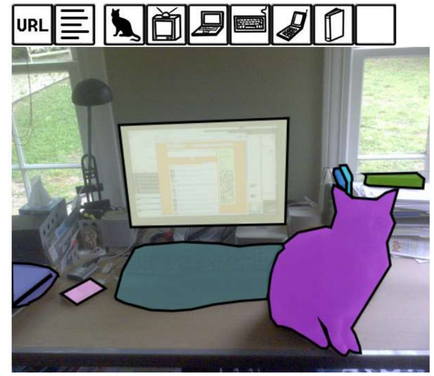
You Only Look Once (YOLO)

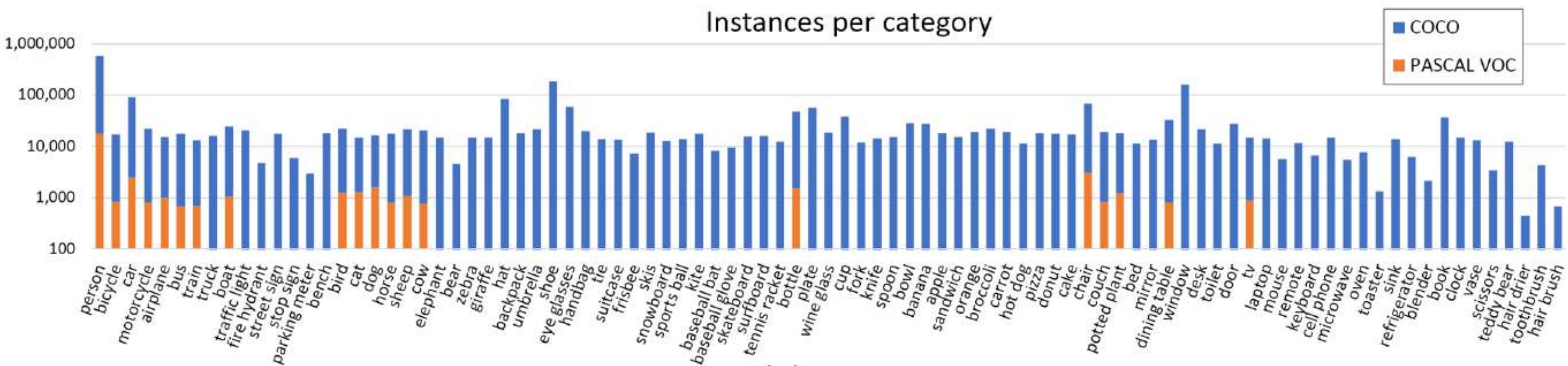
Georgia Tech

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)

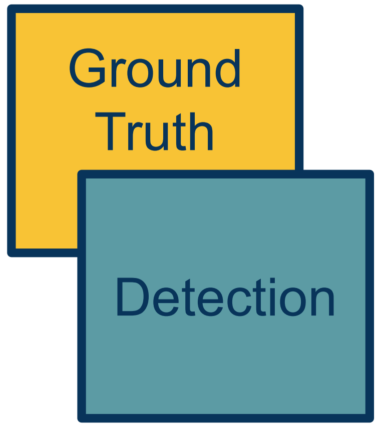
Categories per image

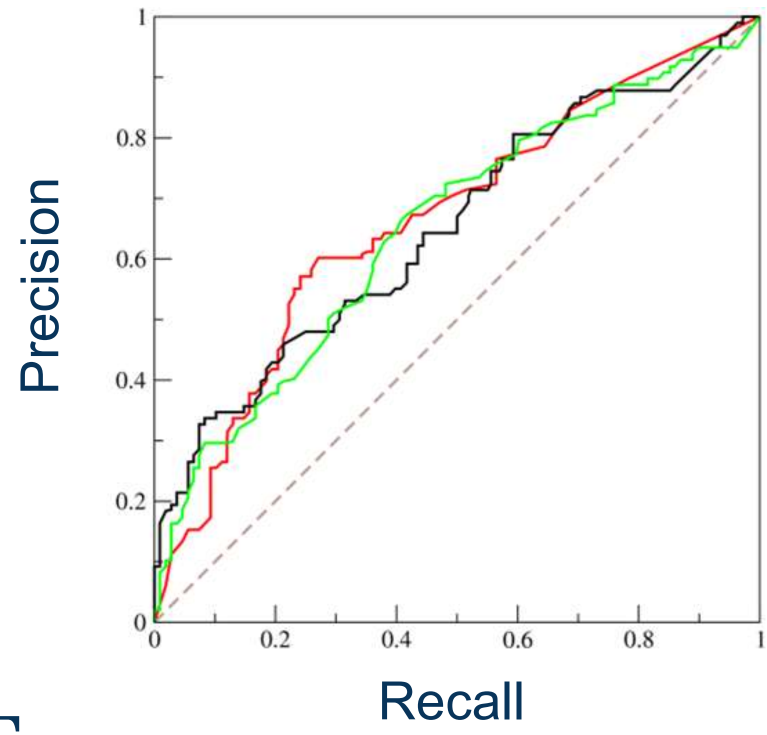
Instances per image

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

Datasets

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)

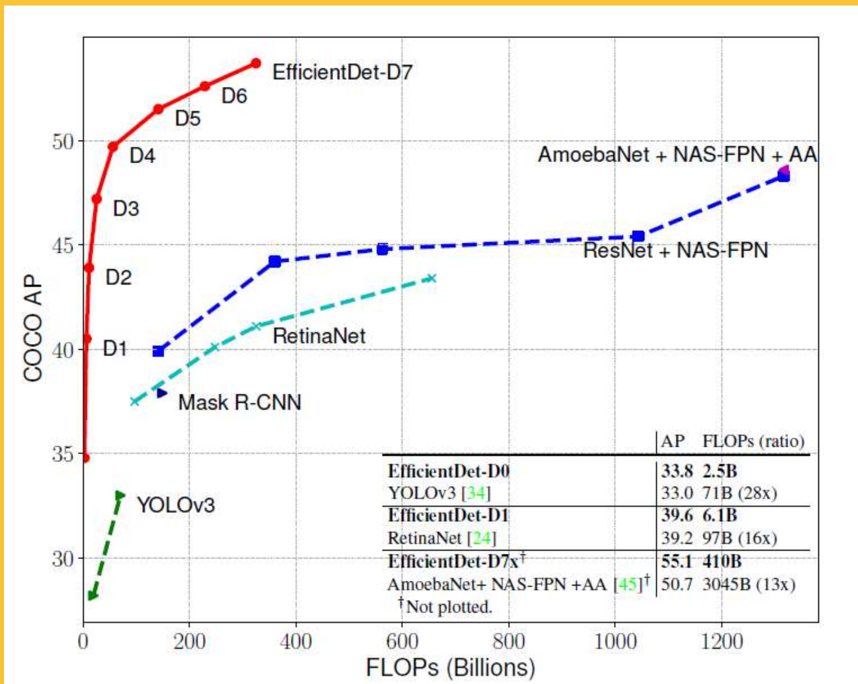




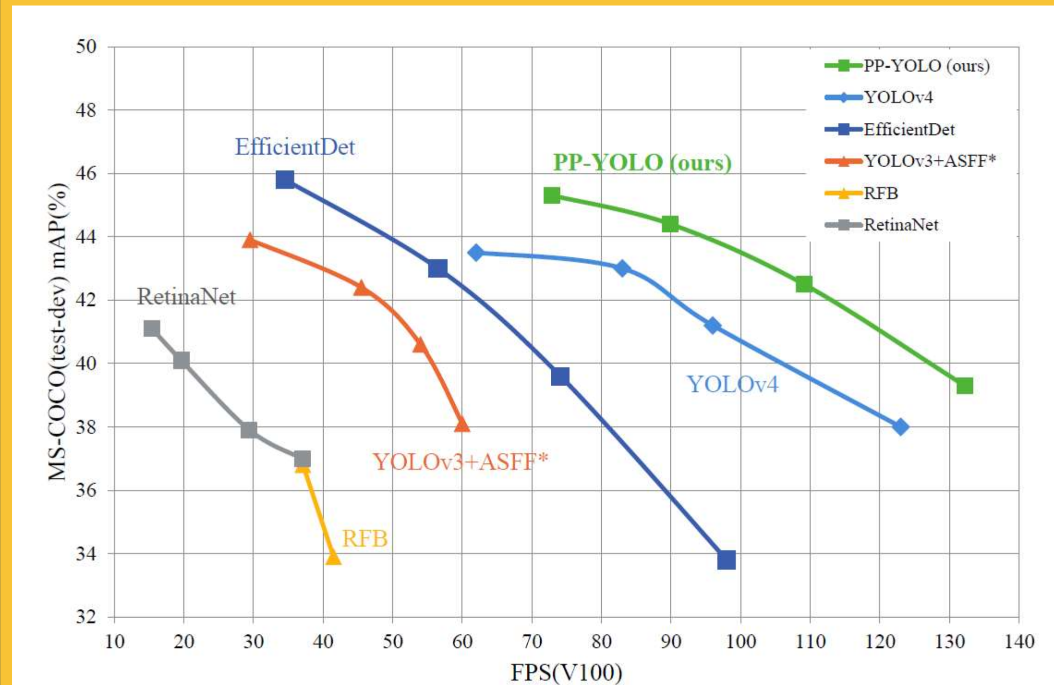
$$\boldsymbol{mAP = \frac{1}{11} \sum_{i \in [0, 0.1, \ldots 1.0]} AP_i}$$

Evaluation – Mean Average Precision (mAP)

Results



EfficientDet

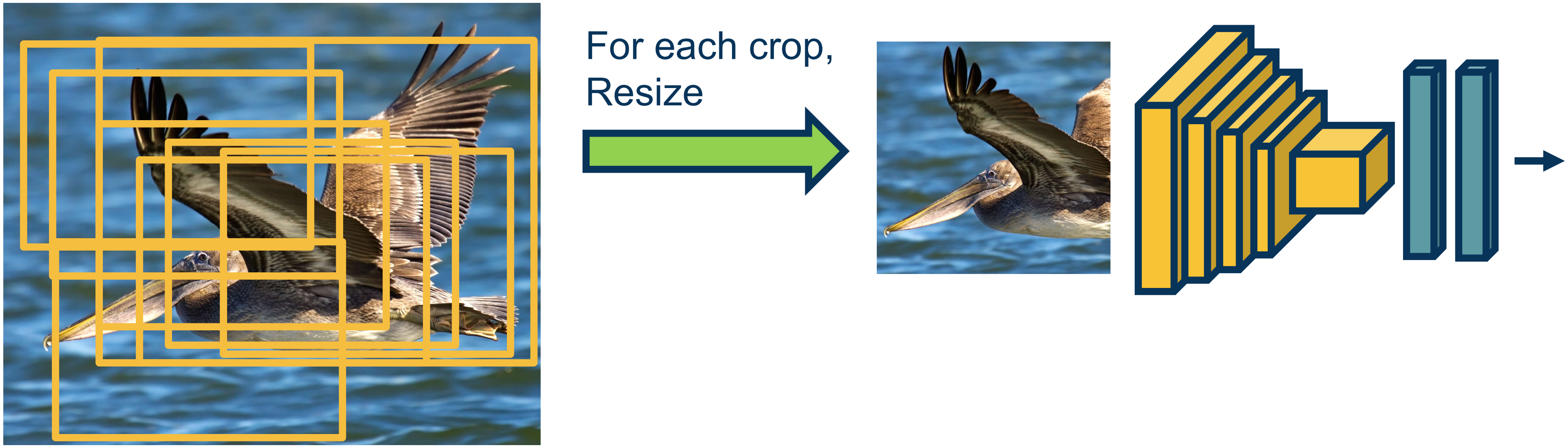


PP-YOLO

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

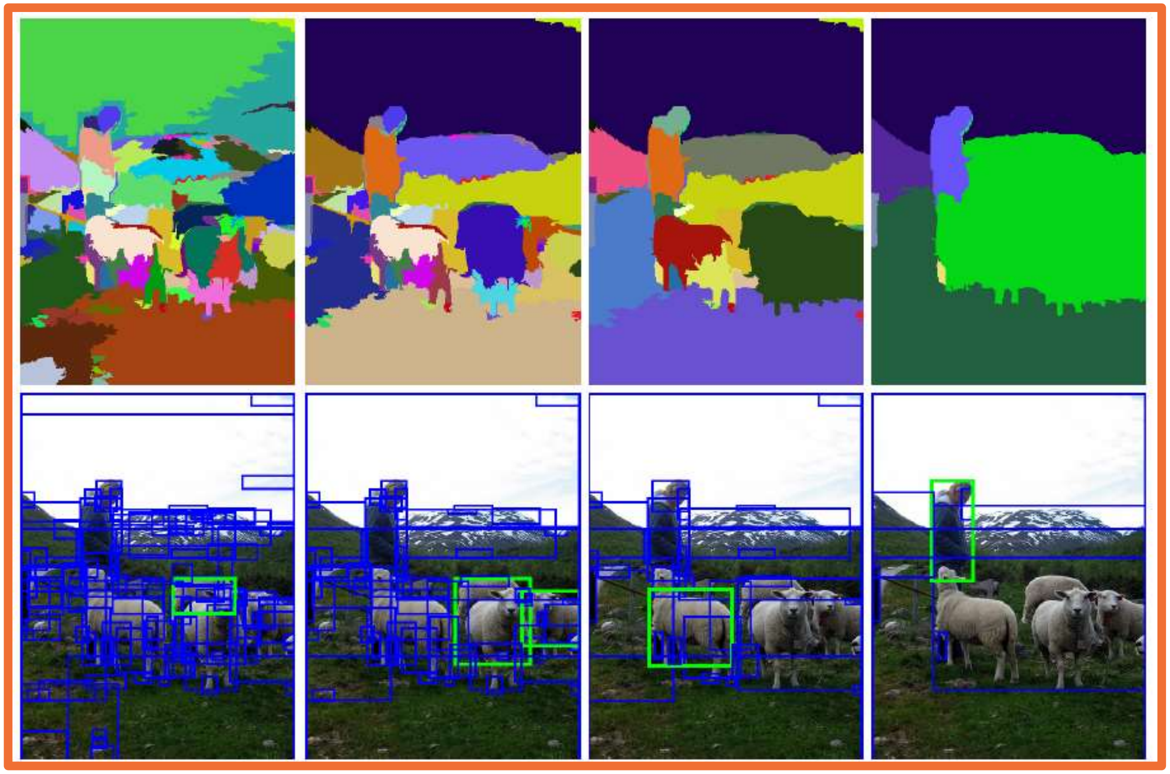
R-CNN

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

Downsides:

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

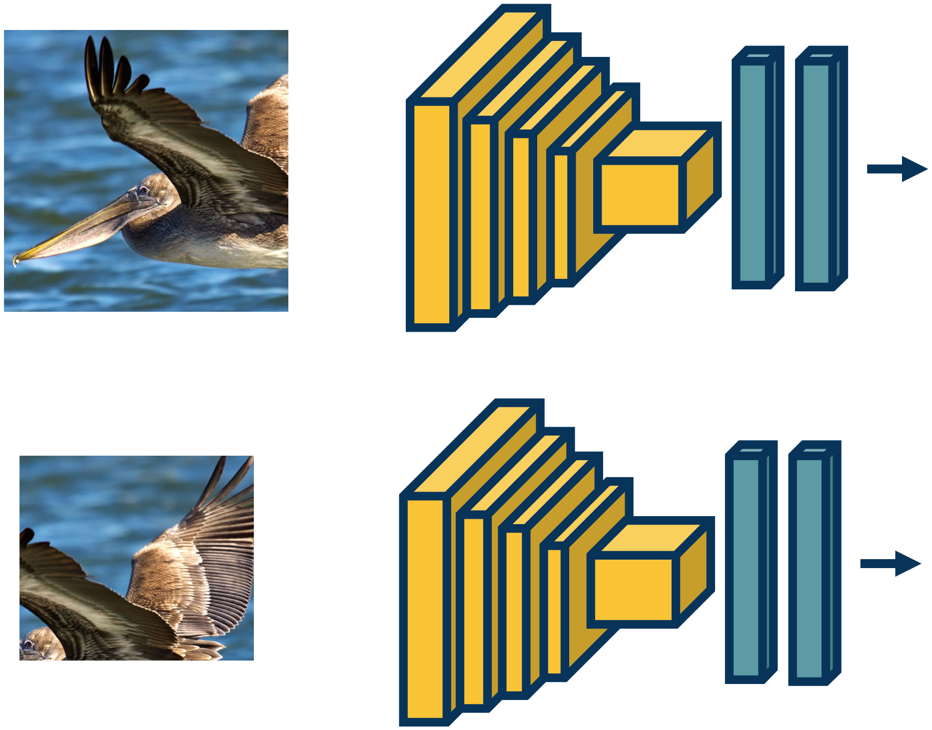
Resize each candidate to full input size and classify



Uijlings, et al., "Selective Search for Object Recognition", 2012

Extracting Region Proposal

What is the problem with this?

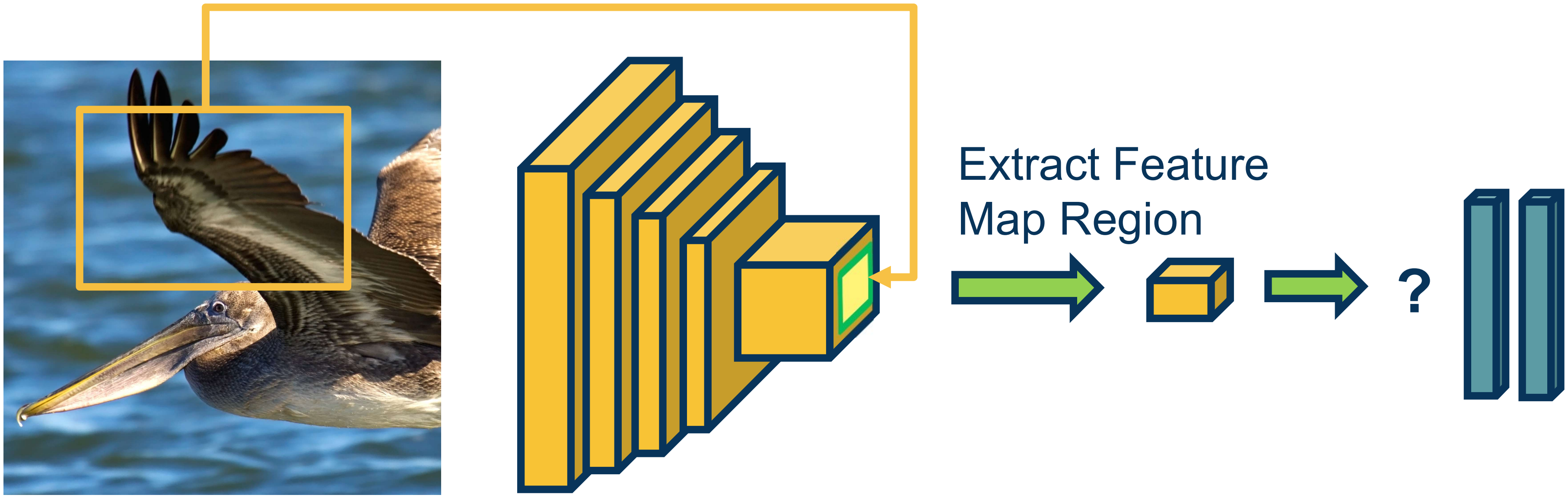


Computation for convolutions re-done for each image patch, even if overlapping!

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

Inefficiency of R-CNN

Map each ROI in image to corresponding region in feature maps

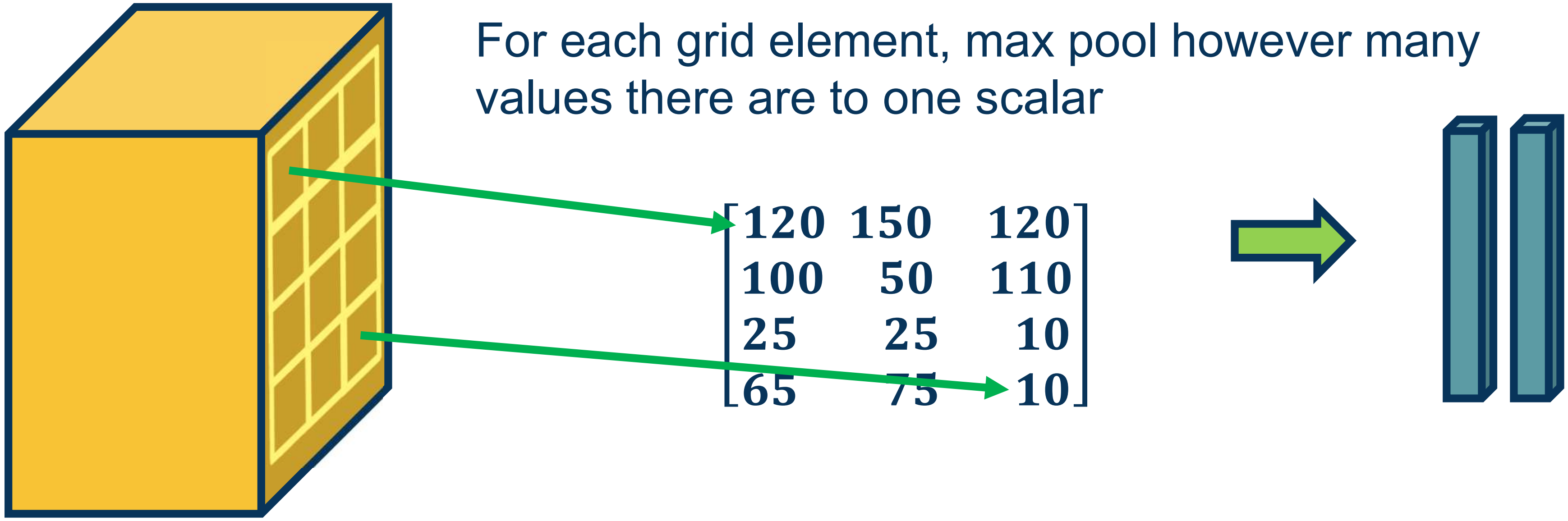


Idea: ***Reuse*** computation by finding regions in ***feature maps***

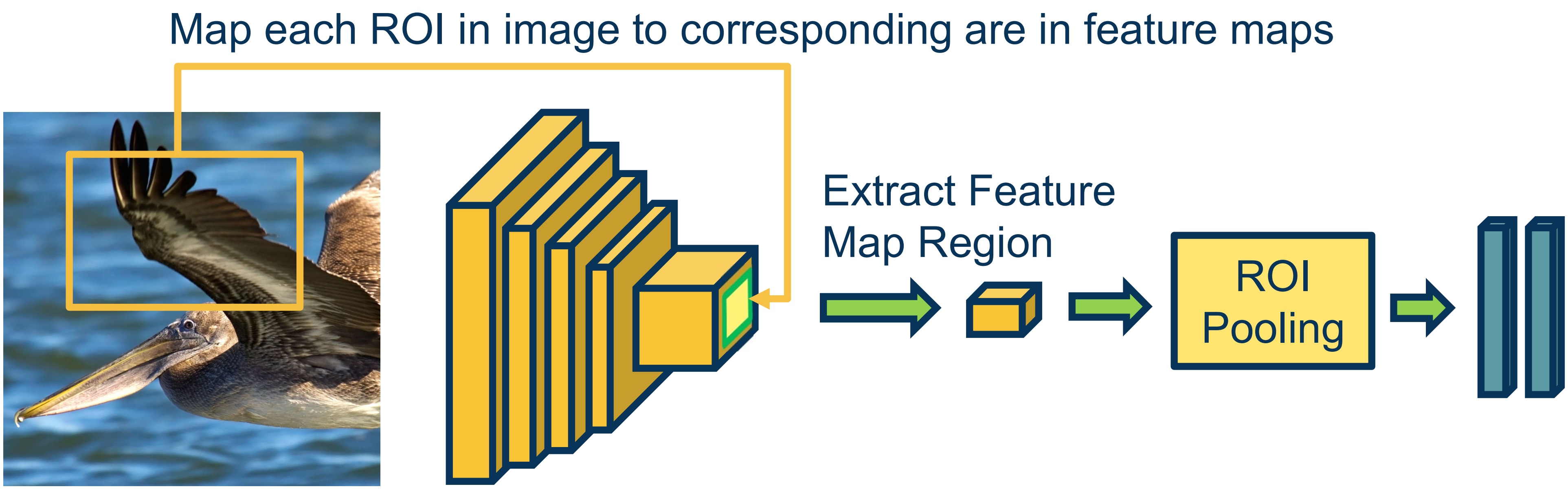
- Feature extraction only done once per image now!
- Problem: Variable input size to FC layers (different feature map sizes)

Girshick, "Fast R-CNN", 2015

ROI Pooling



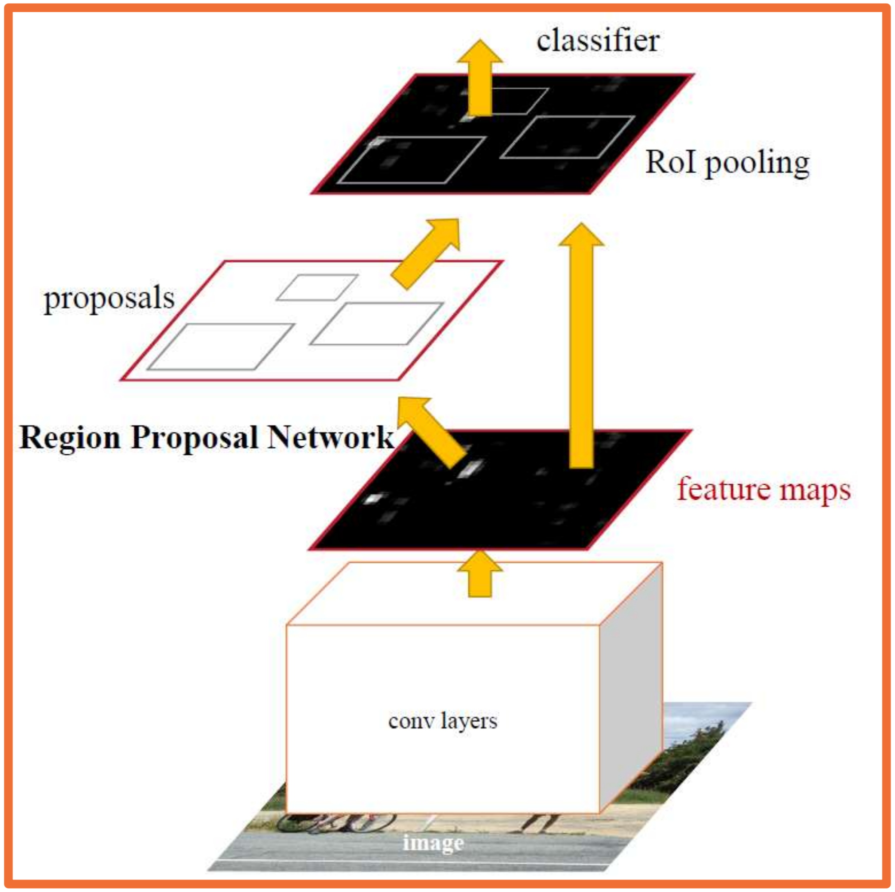
Given an arbitrarily-sized feature map, we can use **pooling** across a grid (ROI Pooling Layer) to convert to fixed-sized representation



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

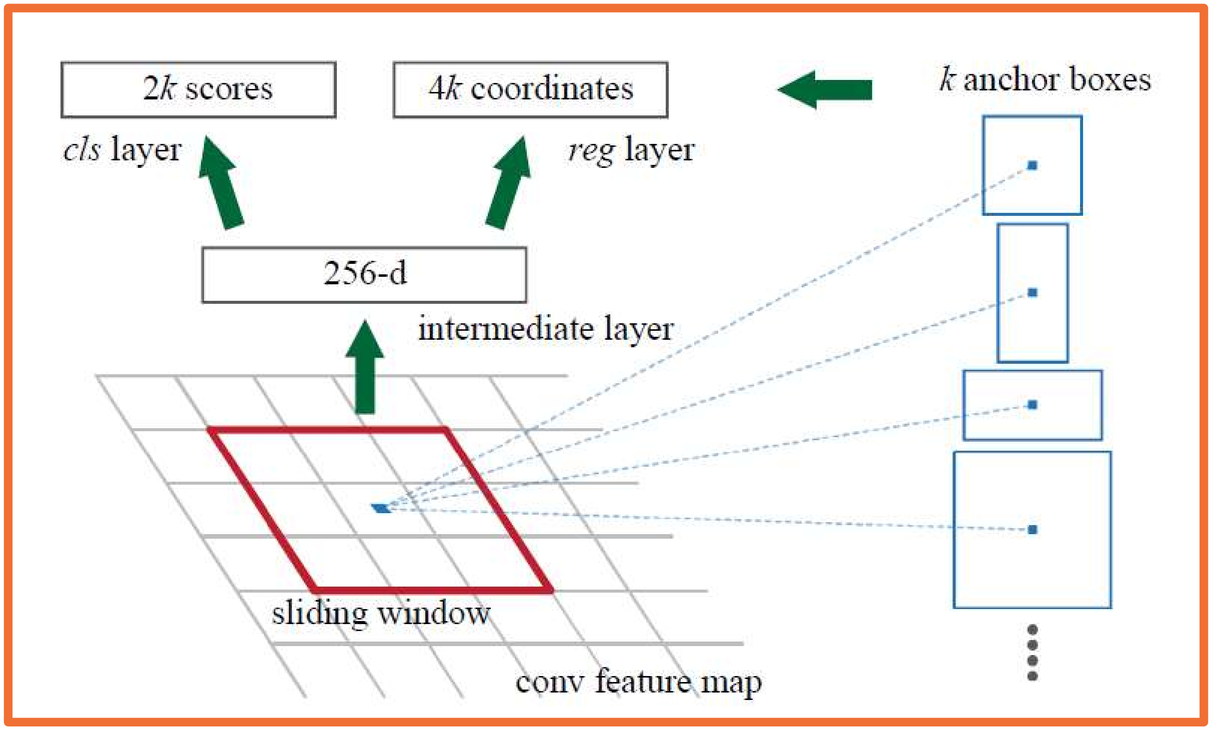


Ren, et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", 2016

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

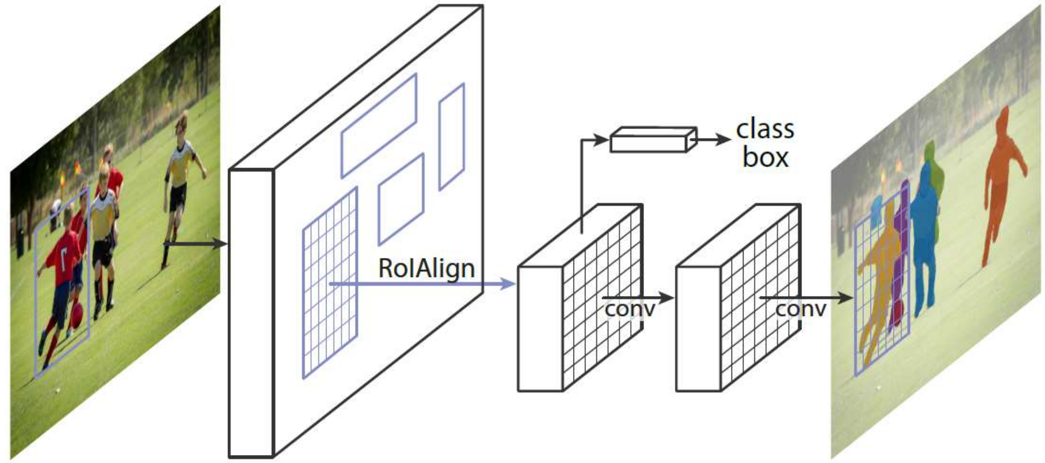


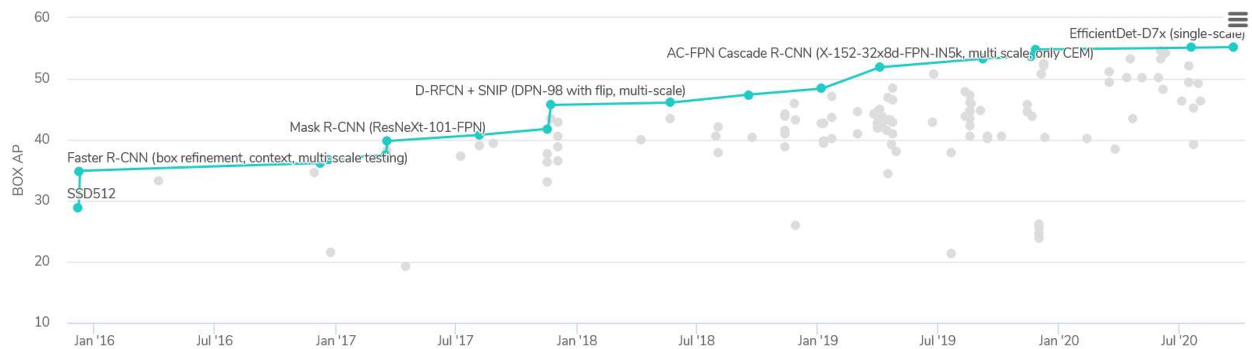
Ren, et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", 2016

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

- A range of problems characterized by **density and type of output**
- **Semantic/instance segmentation:** Dense, spatial output
  - Leverage encoder/decoder architectures
- **Object detection:** Variable-length list of objects
  - Two-stage versus one-stage architectures
  - (Not covered): Anchor-based versus anchor-free methods

Summary

Bias & Fairness

FACEBOOK AI Georgia Tech

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

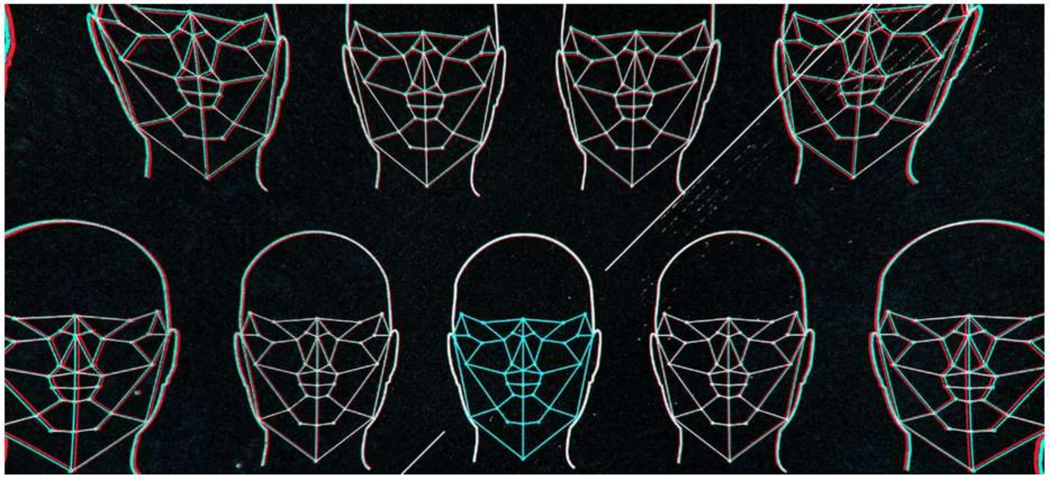
Gender and racial bias found in Amazon's facial recognition technology (again)

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



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

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!**

Calibration

Calibration

- Definition
- Measuring Calibration
- Calibrating models
- Limitations of Calibration

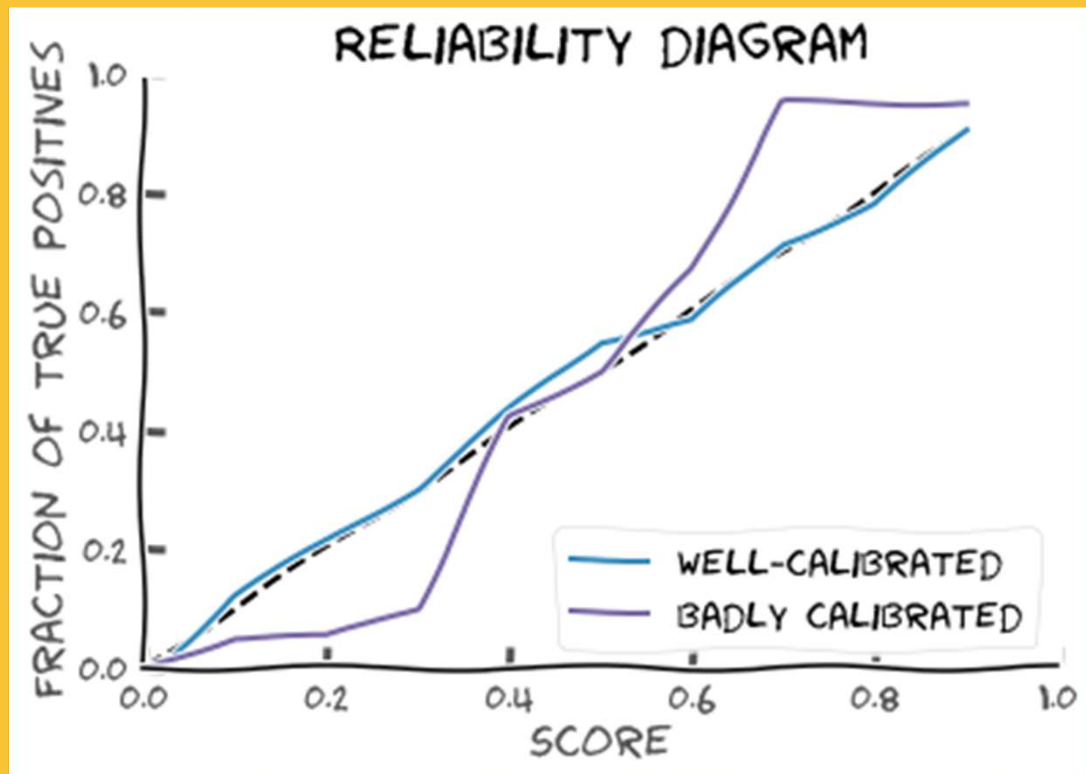
A classifier is **well-calibrated** if the probability of the observations with a given probability score of having a label is equal to the proportion of observations having that label

- **Example:** if a binary classifier gives a score of 0.8 to 100 observations, then 80 of them should be in the positive class

$$\forall p \in [0,1], P(\hat{Y} = Y | \hat{P} = p) = p$$

where $\hat{Y}$ is the predicted label and $\hat{P}$ is the predicted probability (or score) for class Y

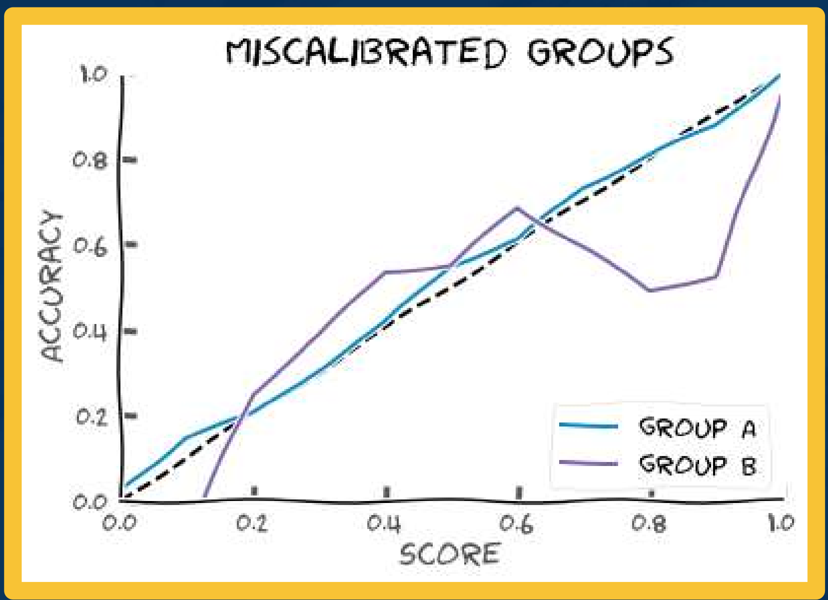
Calibration: Definition

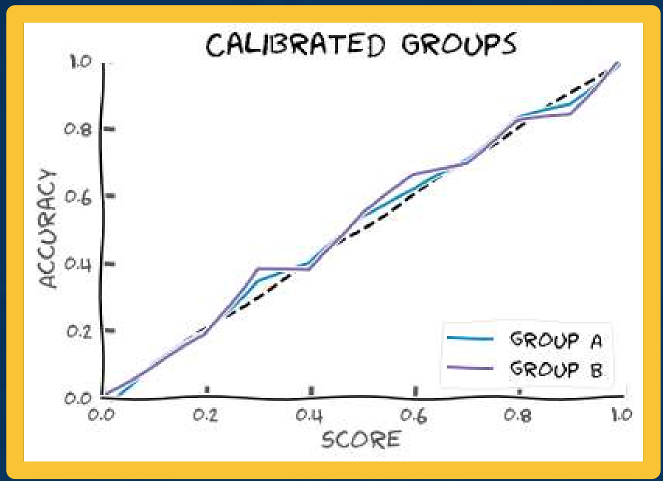


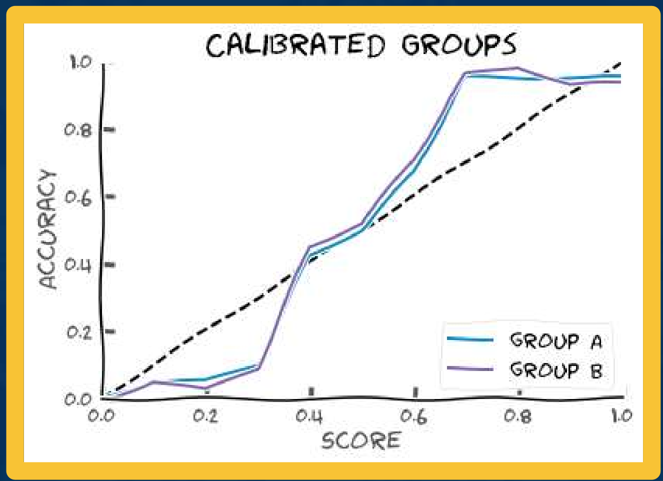
FACEBOOK AI Georgia Tech

Calibration: Definition

Group Calibration: the scores for subgroups of interest are calibrated (or at least, equally mis-calibrated)







FACEBOOK AI Georgia Tech

- Some models (e.g Logistic Regression) tend to have **well-calibrated predictions**
- Some DL models (e.g. ResNet) tend to be **overconfident** (*https://arxiv.org/pdf/1706.04599.pdf*)
- **Logistic calibration/Platt scaling**

Post-processing approach requiring an **additional validation dataset**

Platt scaling (binary classifier)

- Learn parameters $\boldsymbol{a, b}$ so that the **calibrated probability** is $\widehat{\boldsymbol{q}}_{\boldsymbol{i}} = \boldsymbol{\sigma}(\boldsymbol{a}\boldsymbol{z}_{\boldsymbol{i}} + \boldsymbol{b})$)where $\boldsymbol{z}_{\boldsymbol{i}}$ is the network's logit output)

Temperature scaling extends this to multi-class classification

- Learn a temperature $\boldsymbol{T}$, and produce calibrated probabilities $\widehat{\boldsymbol{q}}_{\boldsymbol{i}} = \max_{\boldsymbol{k}} \boldsymbol{\sigma}_{\boldsymbol{SoftMax}}(\boldsymbol{z}_{\boldsymbol{i}}/\boldsymbol{T})$

Calibration: Limitations

- Group based
- The Inherent Tradeoffs of Calibration