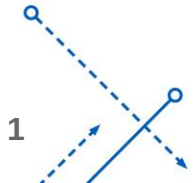
University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

CSE 4/573

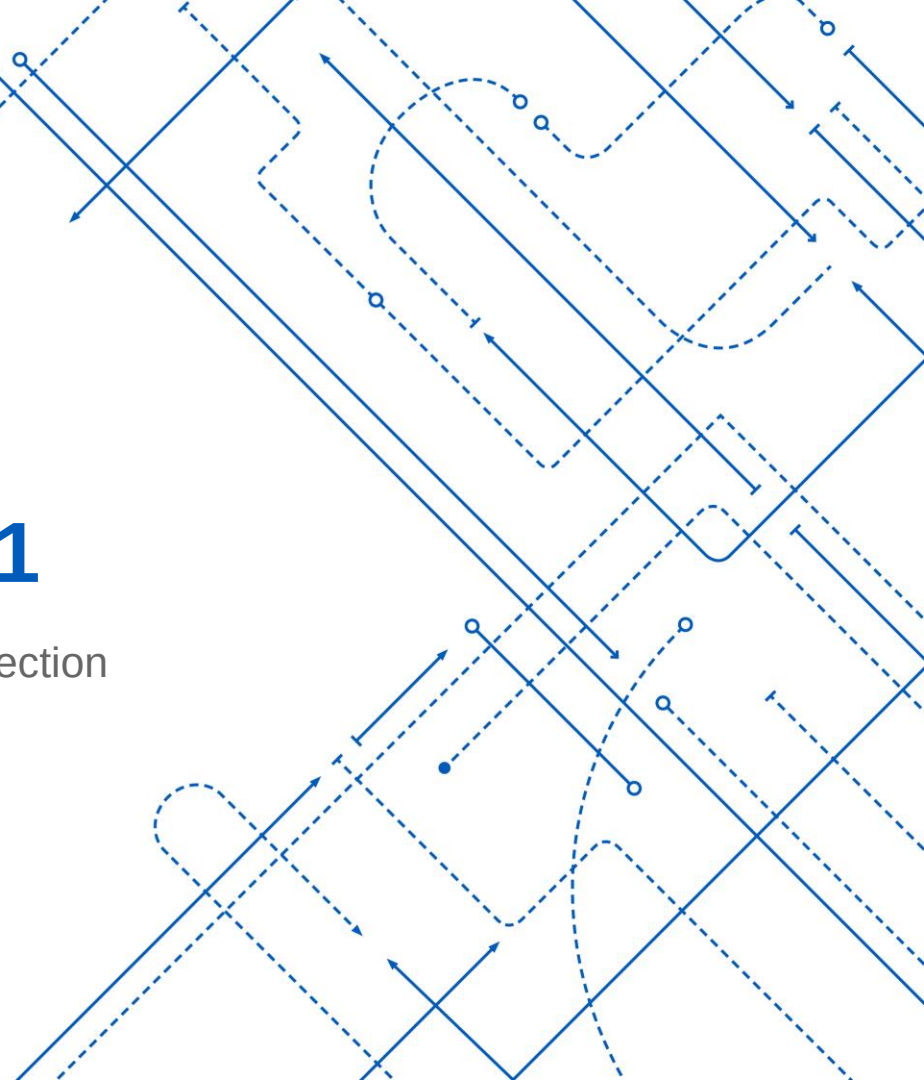
COMPUTER VISION AND IMAGE PROCESSING

Fall 2024

Instructor: Varun Shijo



University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences



MODERN CV TOPICS 1

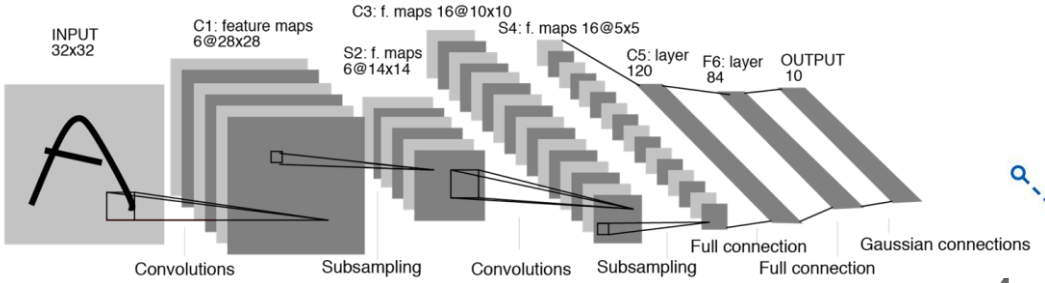
Deep Learning – Segmentation and Object Detection

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

RECAP / BACKGROUND

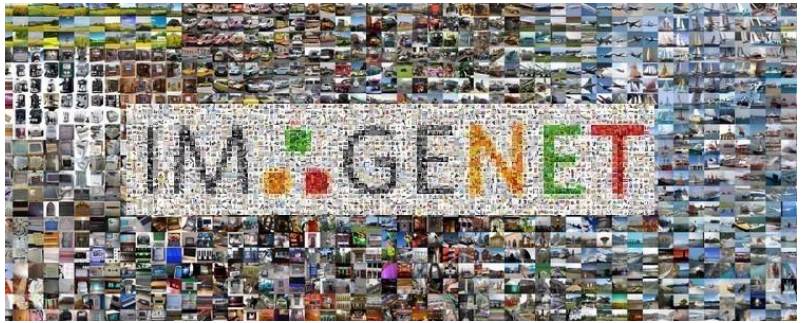
LeNet: First Convnet for Images*

- 99% accuracy on MNIST (Yann LeCun 1998)
- Has all elements of modern convnet
  - Convolutions, maxpooling, fully connected layers
  - Logistic activations after pooling layers (nowadays use RELU)
  - Weight updates through backpropagation



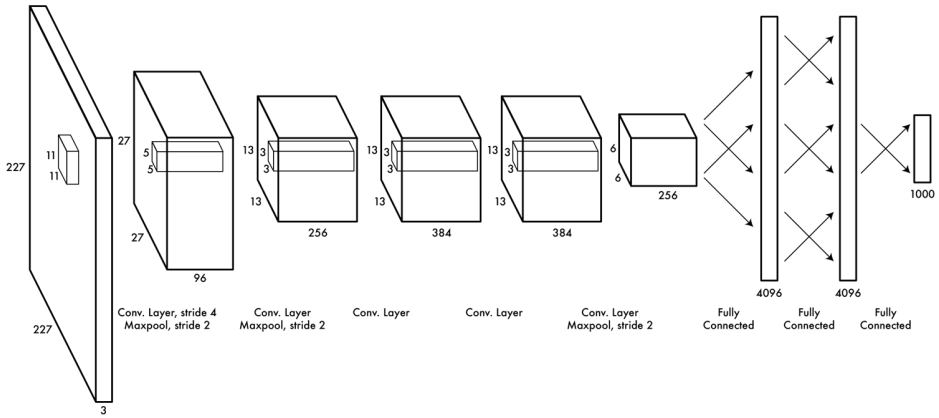
ImageNet: Really big image dataset

- Dataset of images and labels from Princeton
- 14 million images, 22k categories
- Challenge subset (most used):
  - 1.2 million images, 1000 categories



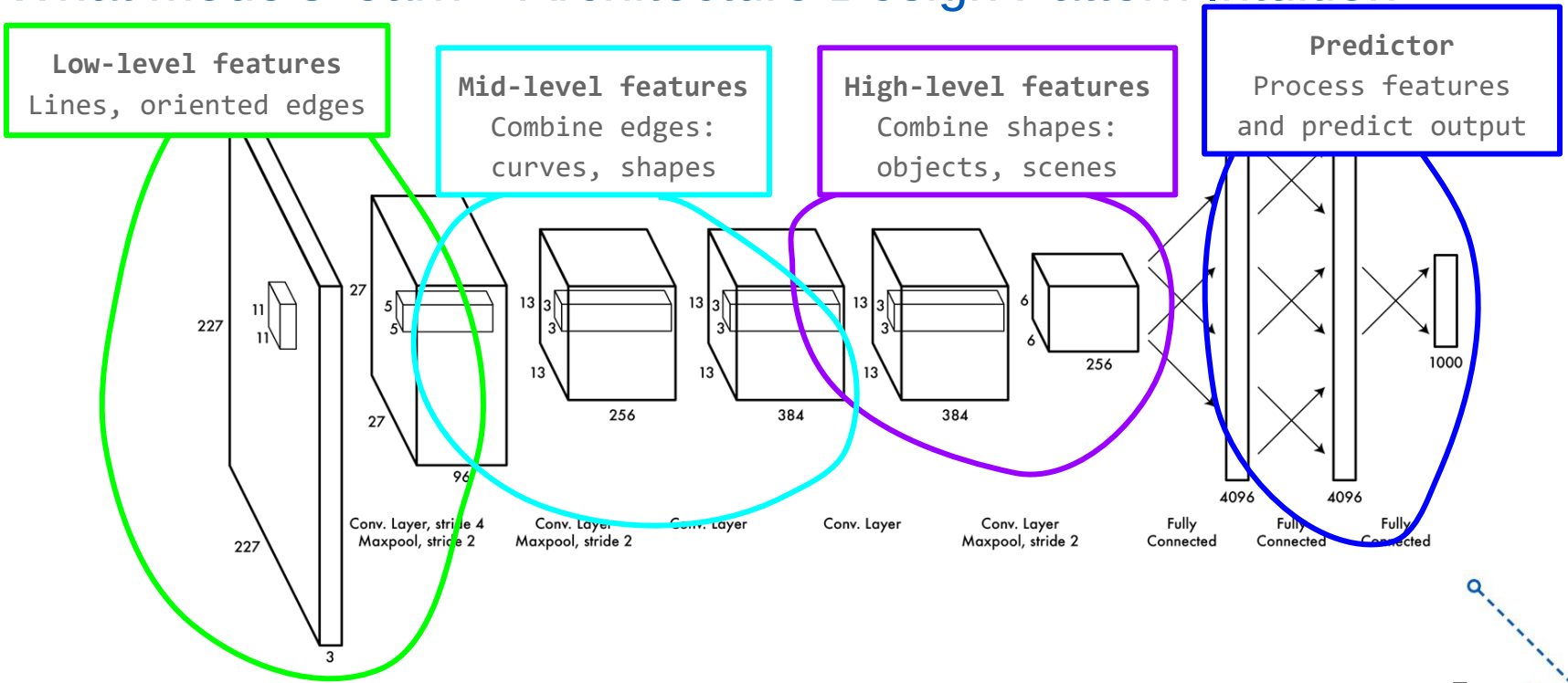
AlexNet: first good network

- Entry for ImageNet challenge
- Much higher accuracy than “traditional” methods (SVM on SIFT)
- Why did it take so long?



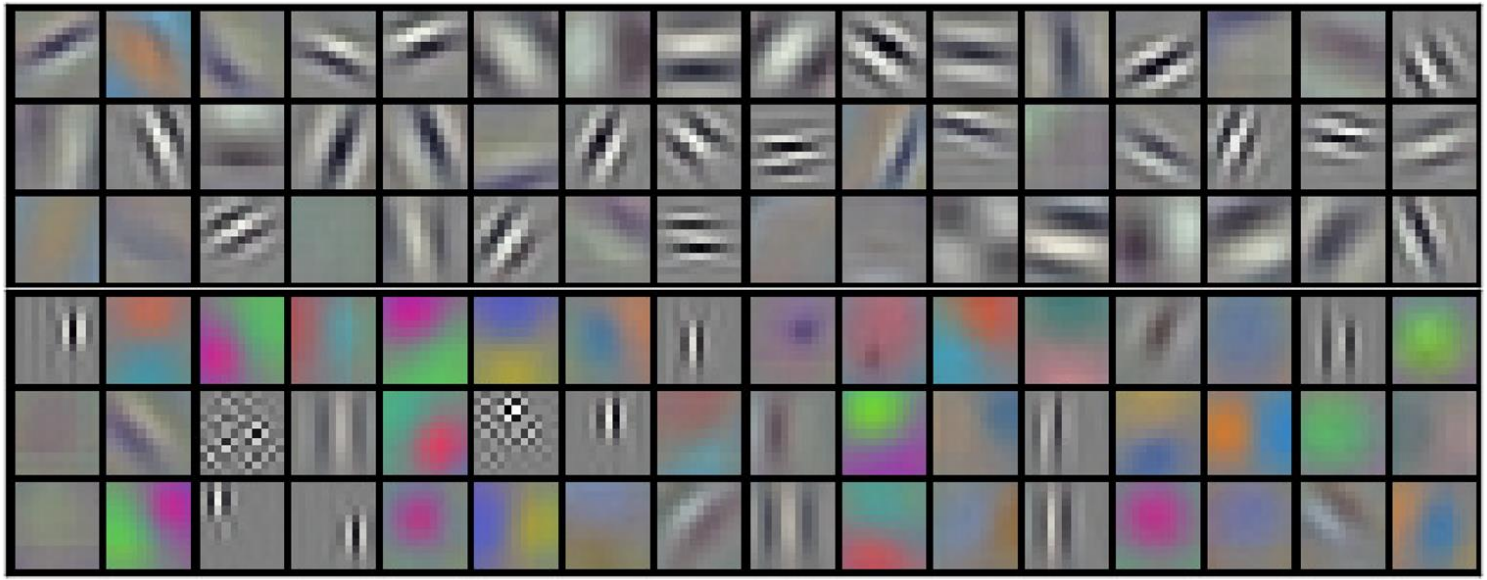
https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

What models learn – Architecture Design Pattern Intuition



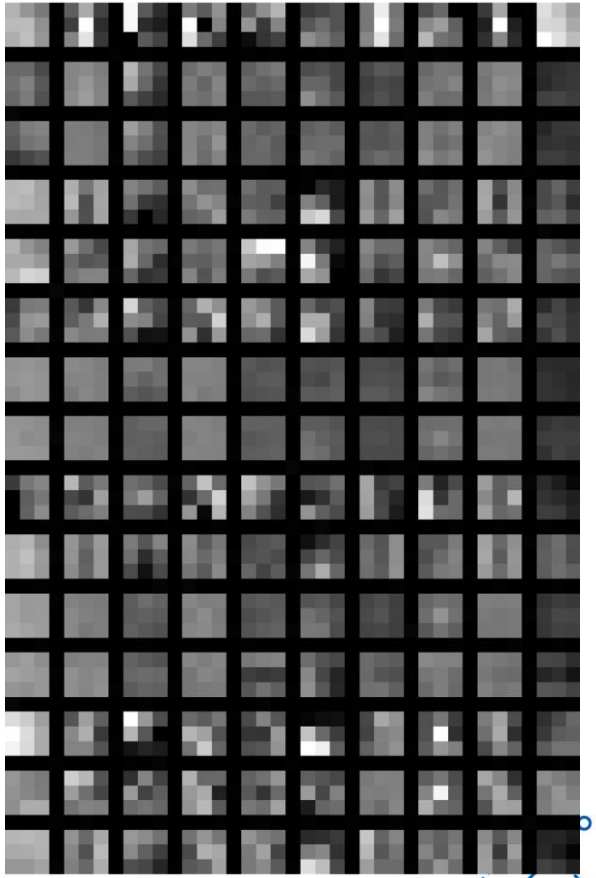
What are these networks learning?

- Features! Here's 1st layer of AlexNet

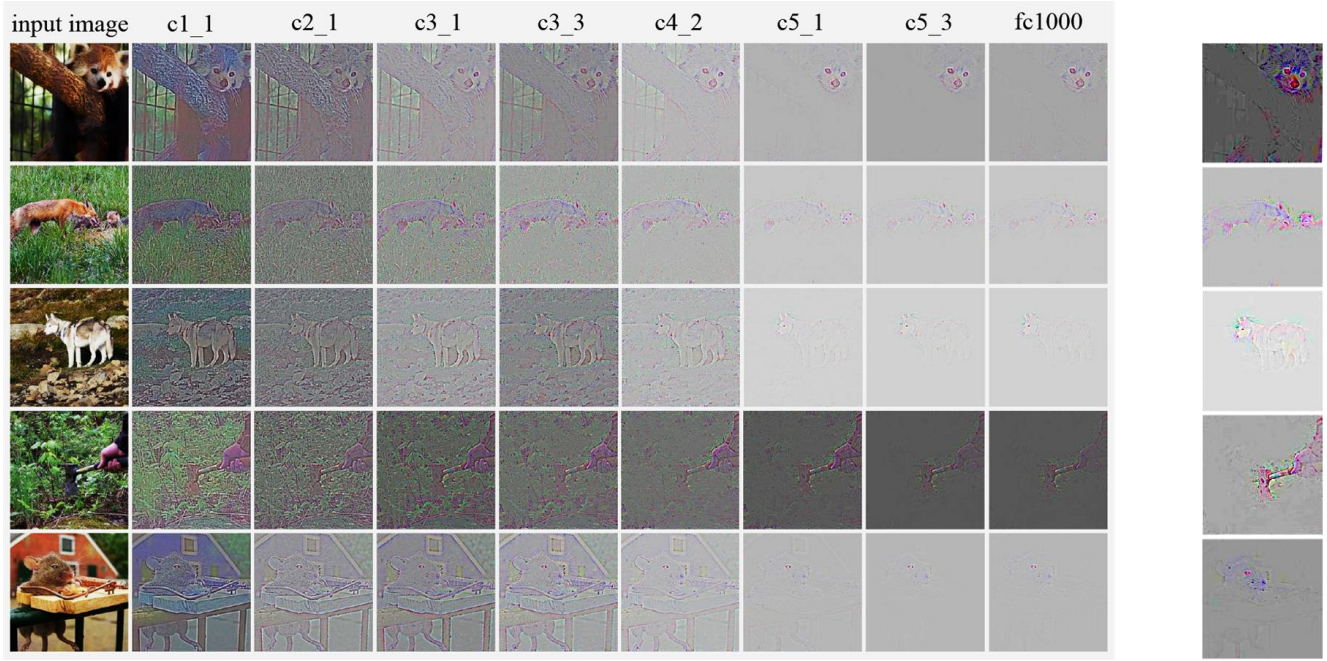


Later layers harder to visualize

- Combinations of lower-level features, not much meaning by themselves
- We can try other methods to visualize them, information flow

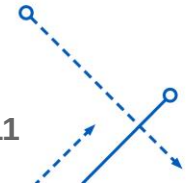


What info goes through the network



Gradient explosion / vanishing

- With very deep networks, the gradients flow through many layers of weights on their way back
- With saturating activation functions like logistic or with small weights gradients can “vanish”
- With non-saturating activations or large weights gradients can EXPLODE!
- Learning doesn’t scale, what works at 2 levels doesn’t at 20



Batch Normalization

- One way to deal with gradient vanishing
  - Normalize activations of filters spatially / over the mini-batch
- During training, the distribution of network activations changes over time because the parameters (weights) change
- Learning is more stable if this change (or *internal covariate shift*) is reduced
- If output is $32 \times 32 \times 16$ image, batch size of 64, normalize activations for each filter across all images in batch: $64 \times 16 \times 32 \times 32$
  - i.e., calculate 16 means and variances
  - Subtract mean, divide by std deviation both across spatial dimensions and images in the batch

https://arxiv.org/pdf/1502.03167.pdf

Batch Normalization

- Other benefits:
  - Output is normalized before activation, mean 0 var 1 means it's in the "good" domain of most activation functions
  - Each image is seen relative to others in a batch, introduces a form of regularization because we don't ever "see" same image twice
  - Stabilizes training so much larger learning rates can be used

https://arxiv.org/pdf/1502.03167.pdf

ResNet: networks making changes

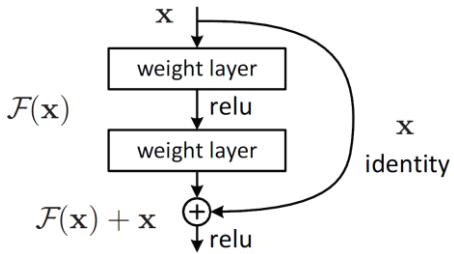
- Normally, output of two layers is:

$$f\big(w * f(vx)\big)$$

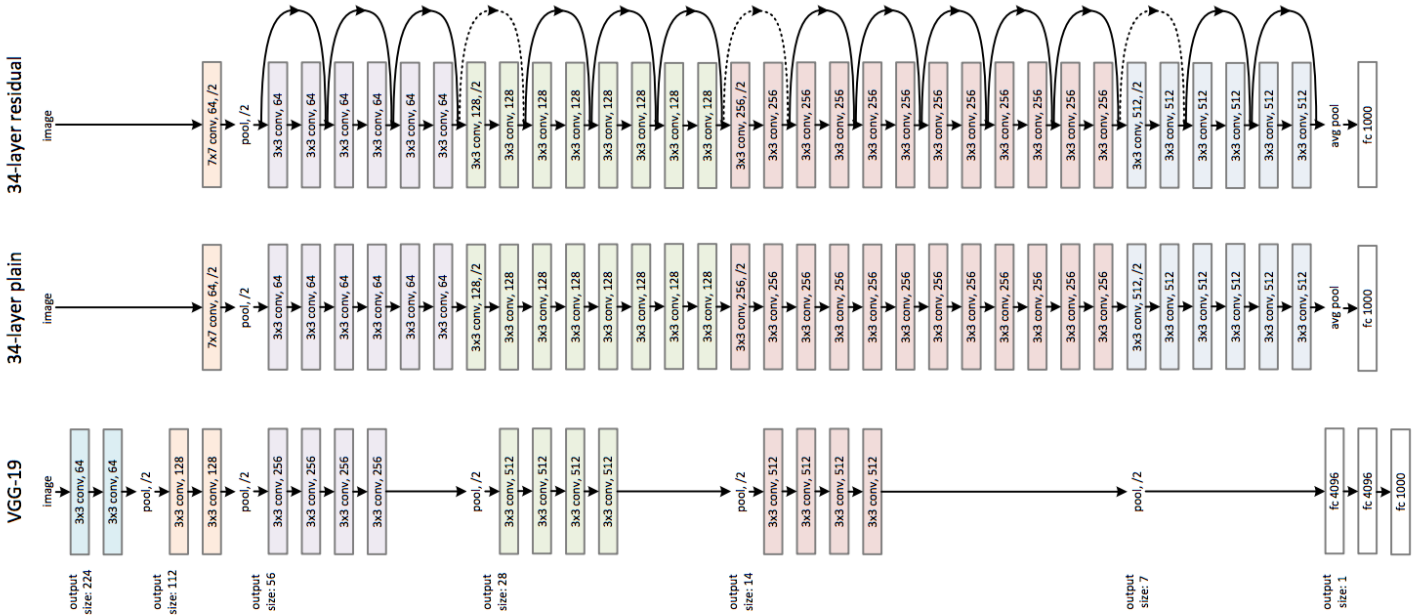
- Residual connections

$$f(w * f(vx) + x)$$

- Learning how to modify x, add some transformed amount
- Gives delta another path, less vanishing gradient



ResNet: networks making changes



ResNet: networks making changes

- 3×3 conv blocks or 3×3 and 1×1 conv blocks
- Residual connections
- VERY deep, 100+ layers

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

UNDERSTANDING ARCHITECTURES

ResNeXt

- Replace 3x3 blocks with larger grouped convs
- “Larger” network but same computational complexity

stage	output	ResNet-50	**ResNeXt-50 (32×4d)**
conv1	112×112	7×7, 64, stride 2	7×7, 64, stride 2
conv2	56×56	3×3 max pool, stride 2 $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	3×3 max pool, stride 2 $\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128, C{=}32 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3	28×28	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256, C{=}32 \\ 1\times1, 512 \end{bmatrix}\times4$
conv4	14×14	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512, C{=}32 \\ 1\times1, 1024 \end{bmatrix}\times6$
conv5	7×7	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 1024 \\ 3\times3, 1024, C{=}32 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	global average pool 1000-d fc, softmax	global average pool 1000-d fc, softmax
# params.		**25.5**×10^6	**25.0**×10^6
FLOPs		**4.1**×10^9	**4.2**×10^9

https://arxiv.org/pdf/1611.05431.pdf

Grouped convolutions

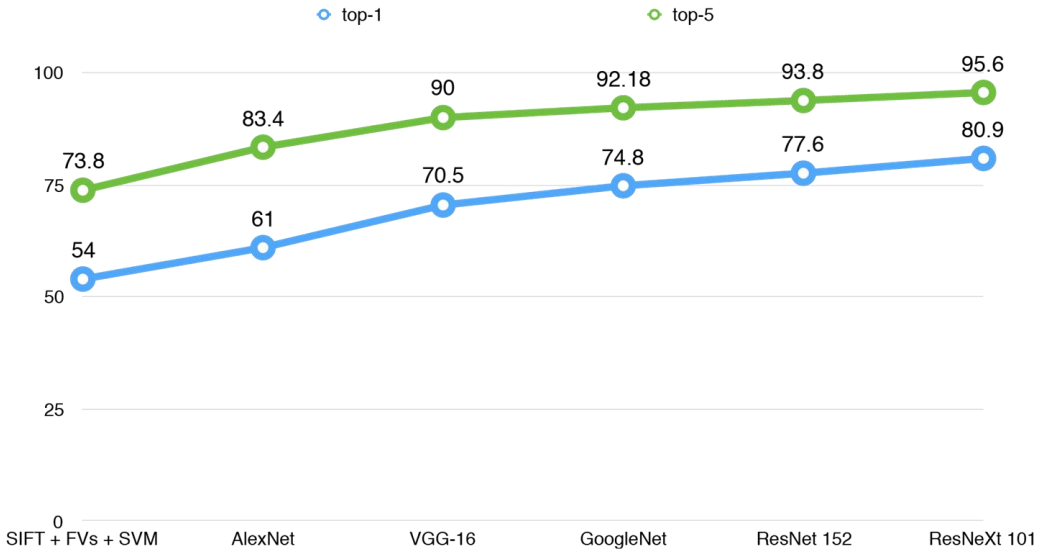
- Most filters look at every channel in input
  - Very expensive
  - Maybe not needed? Might only pull info from a few of them
- Grouped convolutions:
  - Split up input feature map into groups
  - Run convs on groups independently
  - Recombine

Grouped convolutions

- E.g. 3x3 conv layer 32 x 32 x 256 input, 128 filters, 32 groups:
  - Split input into 32 different feature maps
    - Each is 32 x 32 x 8
    - Run 4 filters, 3x3x8 on each group
    - Merge 4*32 channels back together, get 32 x 32 x 128 output
    - Input, output stays same dimensions, less computation

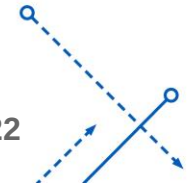
What's NeXt?

- Starting to saturate ImageNet, fighting over 1-2%



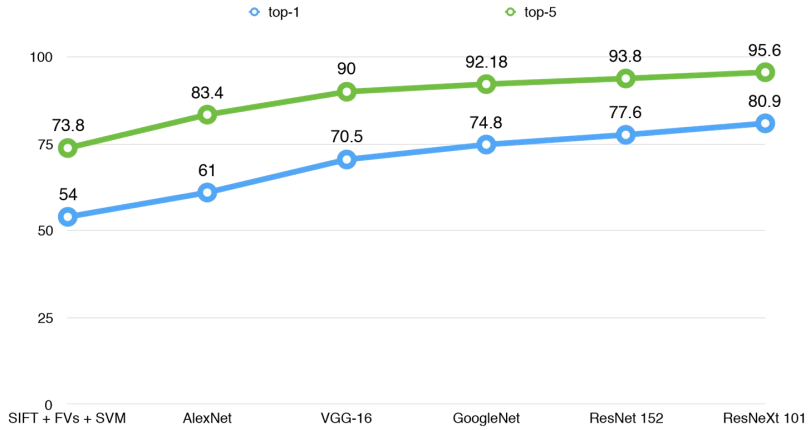
What's NeXt?

- Starting to saturate ImageNet, fighting over 1-2%
- See:
  - https://paperswithcode.com/sota/image-classification-on-imagenet
  - https://arxiv.org/pdf/1905.11946.pdf
  - https://arxiv.org/pdf/1911.04252v1.pdf
- Better training methods (weakly supervised):
  - ResNeXt-101: 85.4%
  - EfficientNet: 87.4%



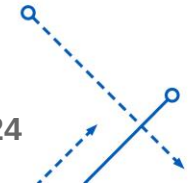
What's NeXt?

- Starting to saturate ImageNet, fighting over 1-2%
- But now vision really *works*, other tasks
  - Other datasets?
  - Segmentation
  - Object detection
  - Captioning
  - …



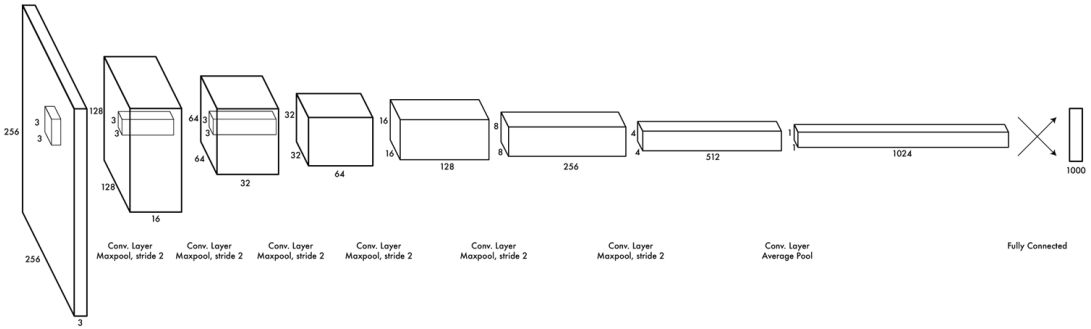
Training classifiers in wild

- Typically you have a much smaller dataset than 1.2 million images and 1,000 classes.
- What problems to do we encounter with less data, say 10,000 images and 500 classes?



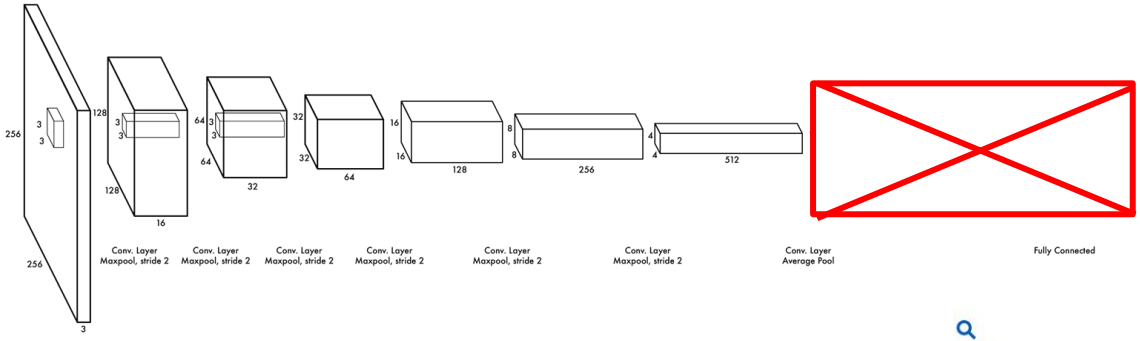
First ImageNet, then the world!

- For dealing with smaller datasets where we might overfit, pretraining is key:
  - First train on ImageNet
  - Chop off last layer
  - Keep training on your data



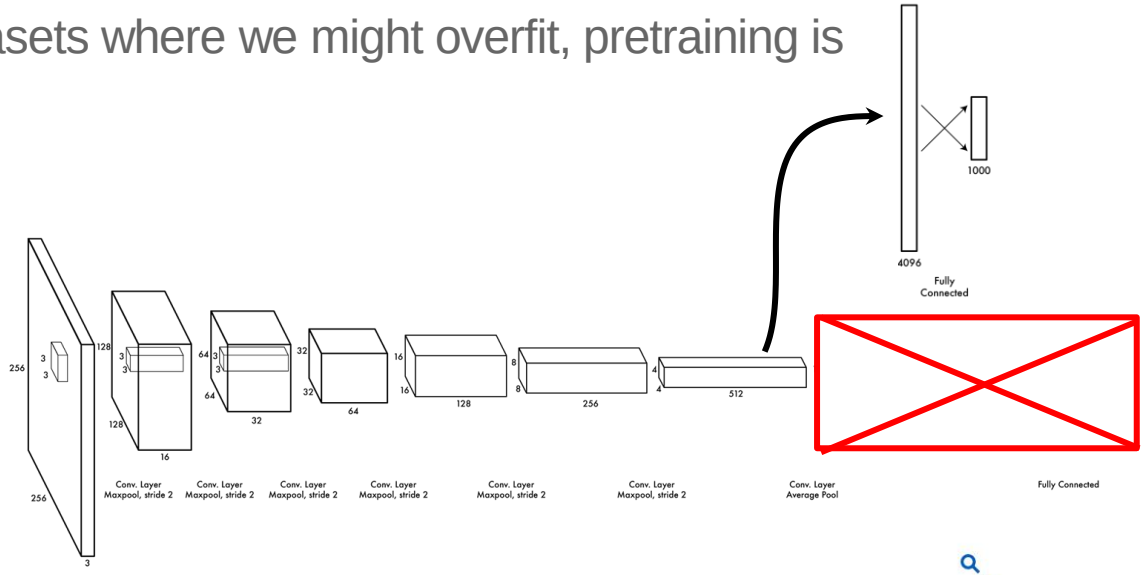
First ImageNet, then the world!

- For dealing with smaller datasets where we might overfit, pretraining is key:
  - First train on ImageNet
  - Chop off last layer
  - Keep training on your data



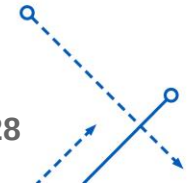
First ImageNet, then the world!

- For dealing with smaller datasets where we might overfit, pretraining is key:
  - First train on ImageNet
  - Chop off last layer
  - Keep training on your data



What's NeXt?

- Starting to saturate ImageNet, fighting over 1-2%
- But now vision really *works*, other tasks
  - Other datasets?
  - Segmentation
  - Object detection
  - Captioning
  - …



What's NeXt?

- Starting to saturate ImageNet, fighting over 1-2%
- But now vision really *works*, other tasks
  - Other datasets?
  - Segmentation
  - Object detection
  - Captioning
  - …

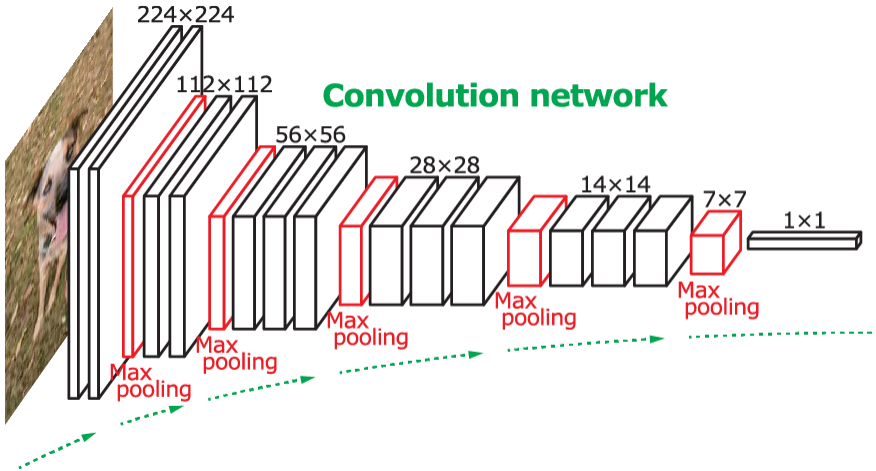
Just different loss functions!

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

SEGMENTATION

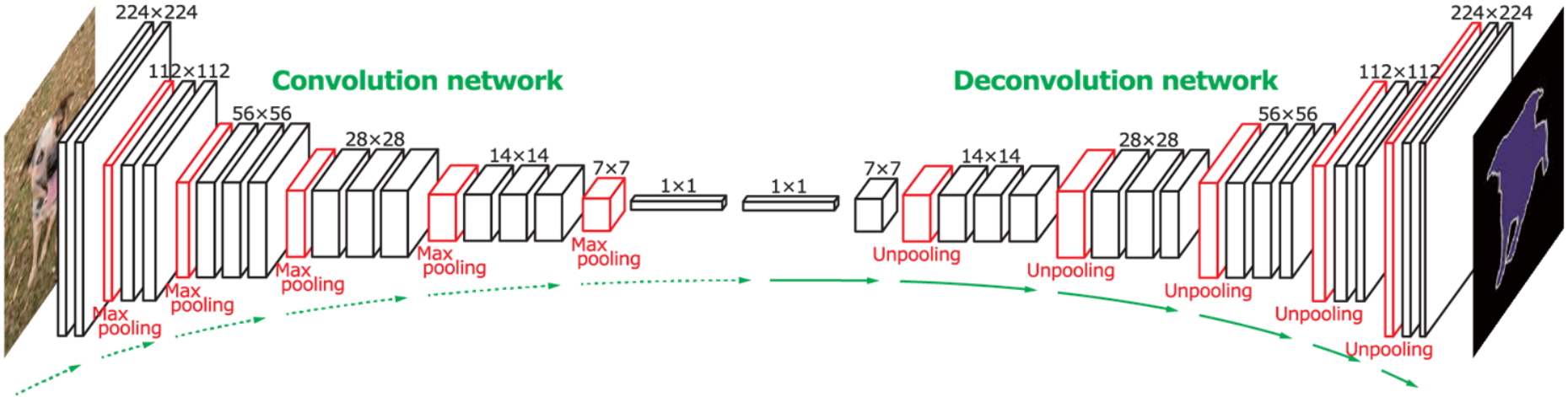
Semantic Segmentation

Semantic Segmentation



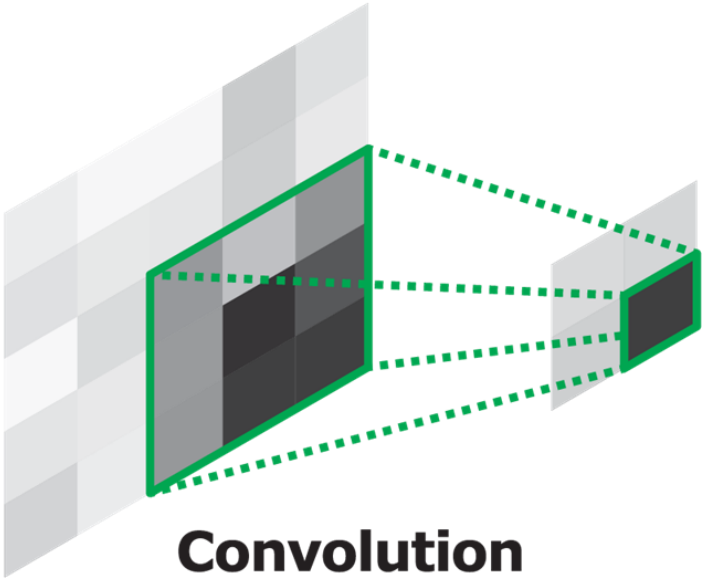
https://arxiv.org/pdf/1505.04366.pdf

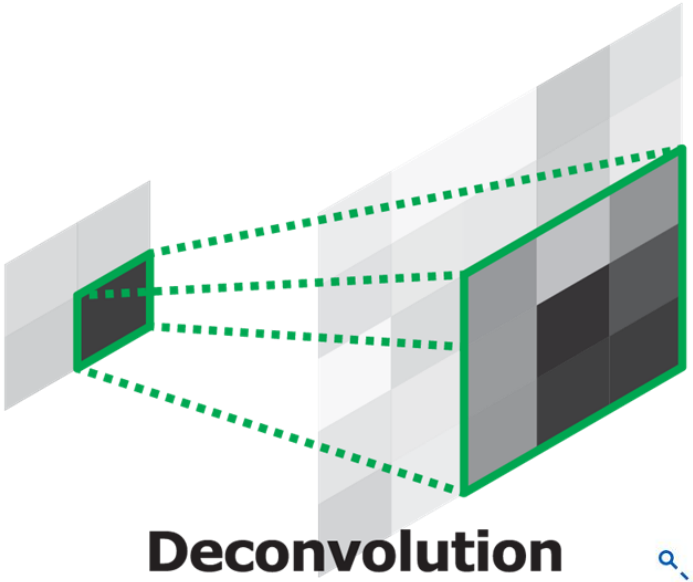
Semantic Segmentation



https://arxiv.org/pdf/1505.04366.pdf

Semantic Segmentation

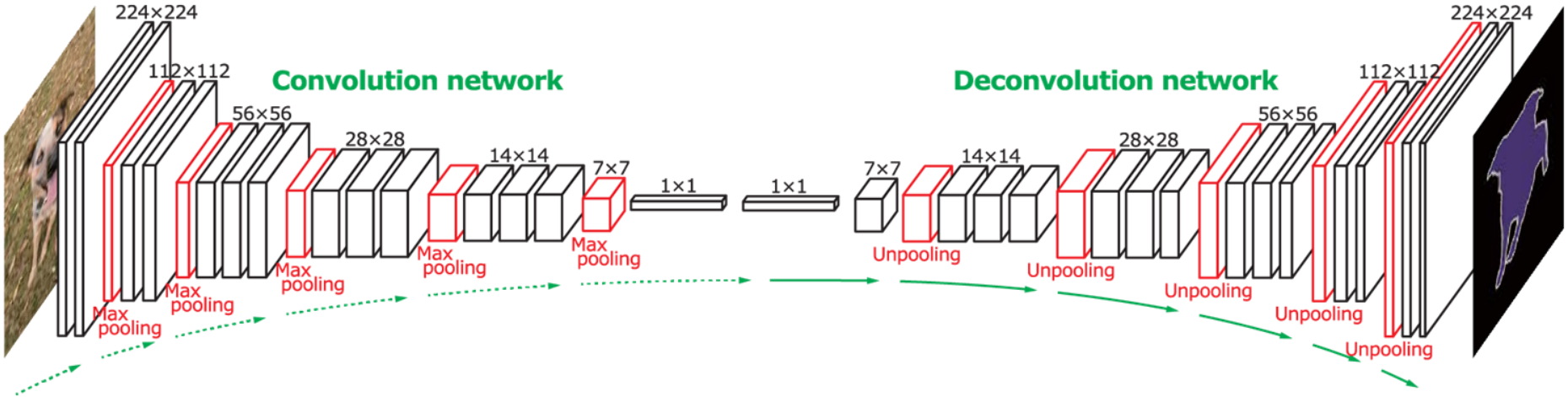




Dr. Joseph Redmon - UW CSE455 - Lec 15

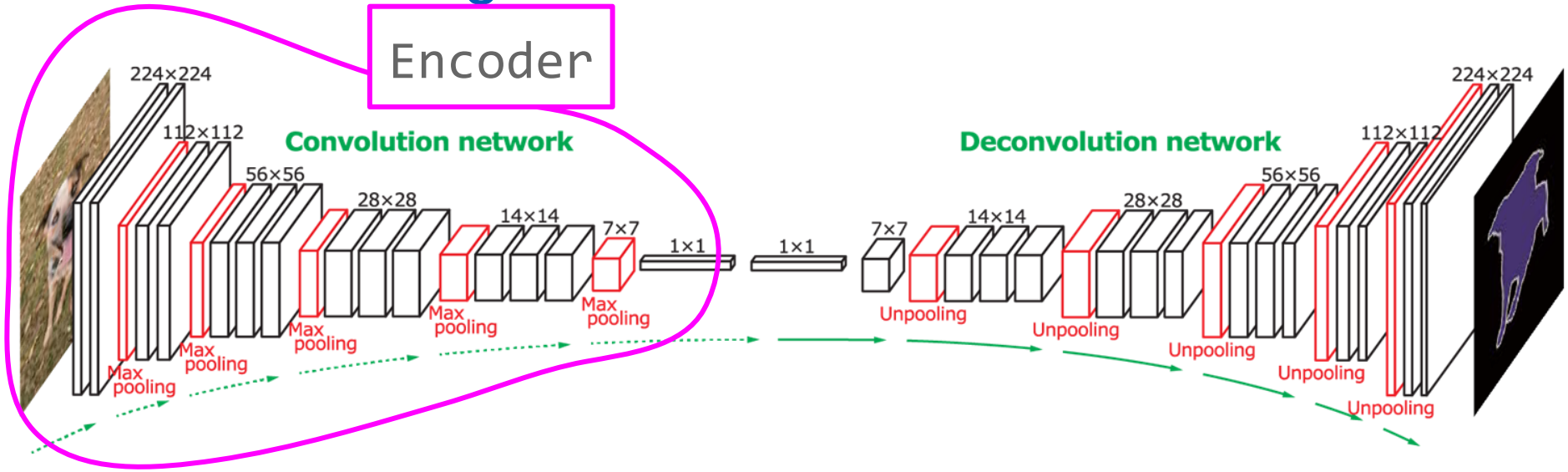
https://arxiv.org/pdf/1505.04366.pdf

Semantic Segmentation

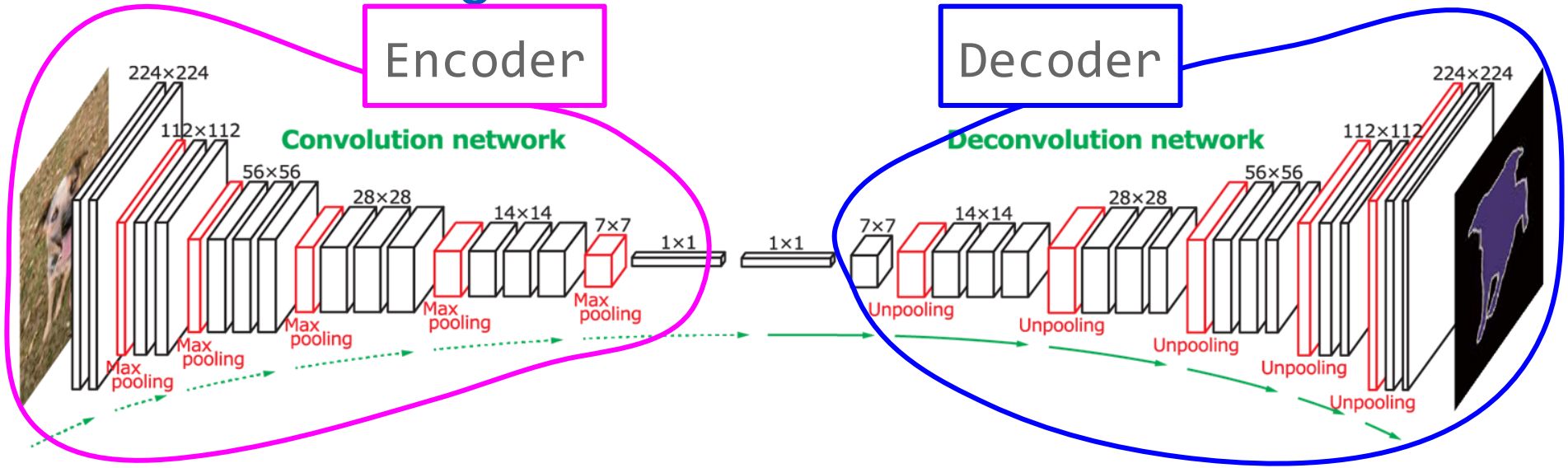


https://arxiv.org/pdf/1505.04366.pdf

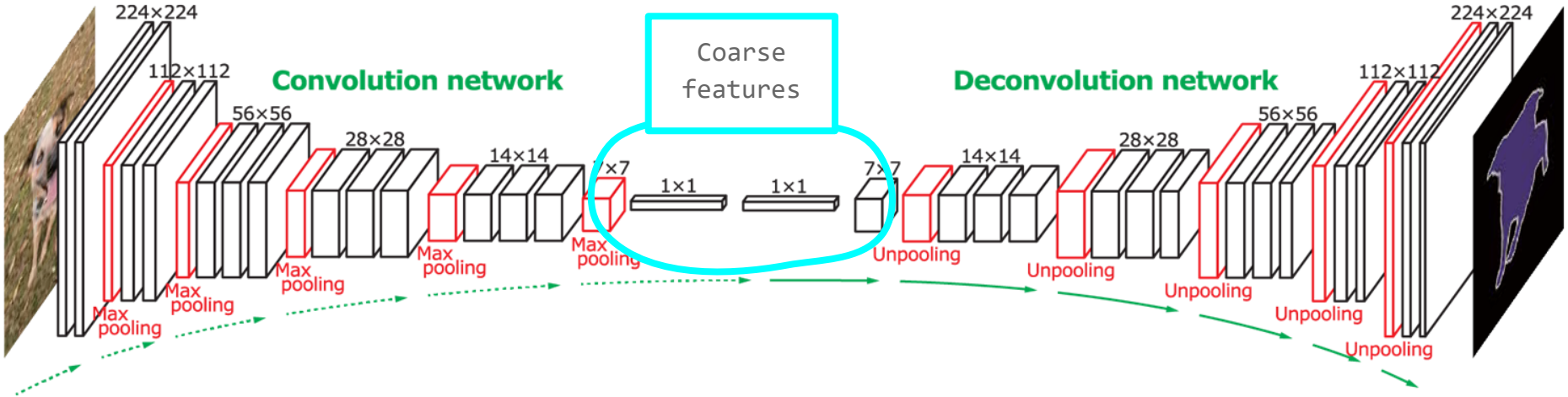
Semantic Segmentation



Semantic Segmentation

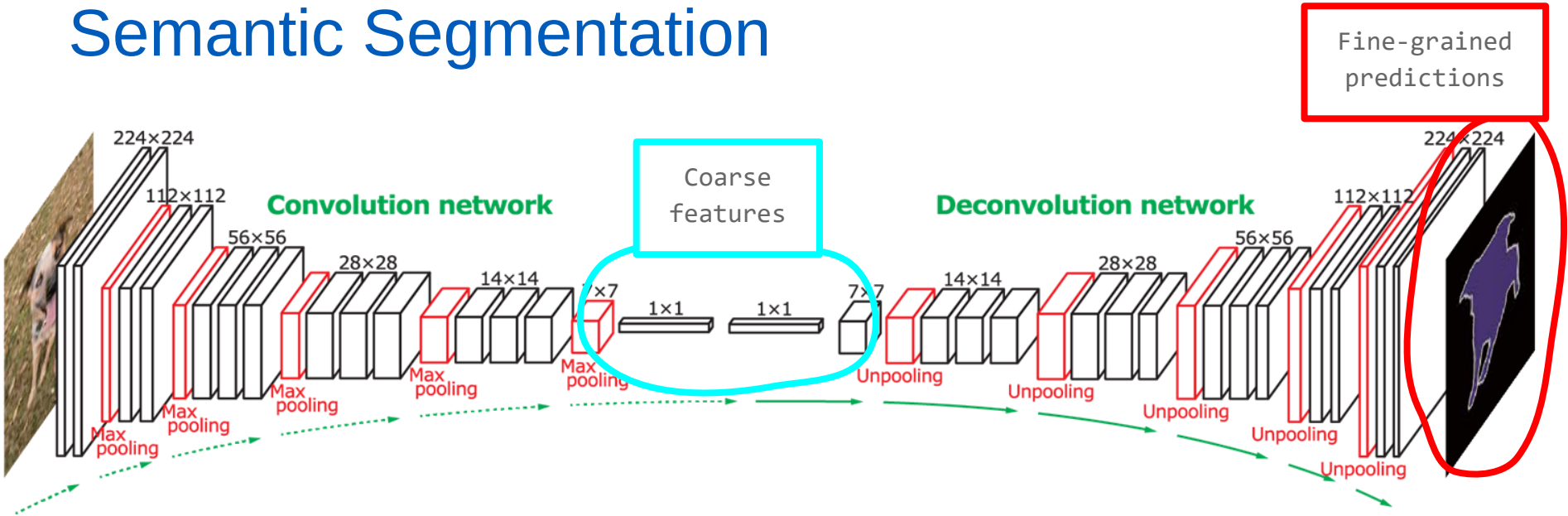


Semantic Segmentation

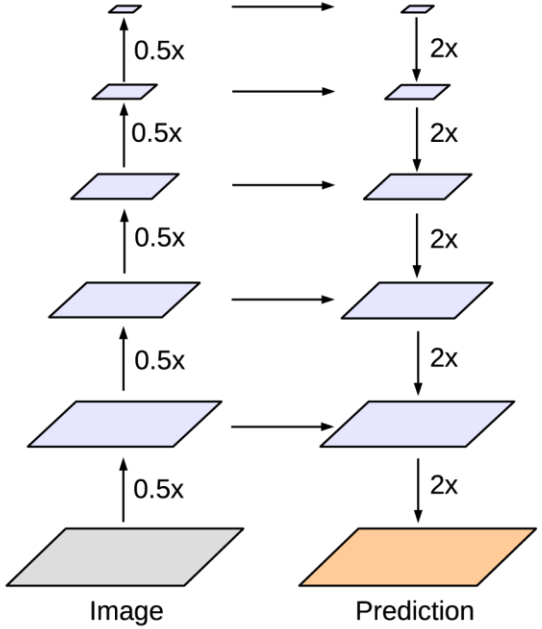


https://arxiv.org/pdf/1505.04366.pdf

Semantic Segmentation

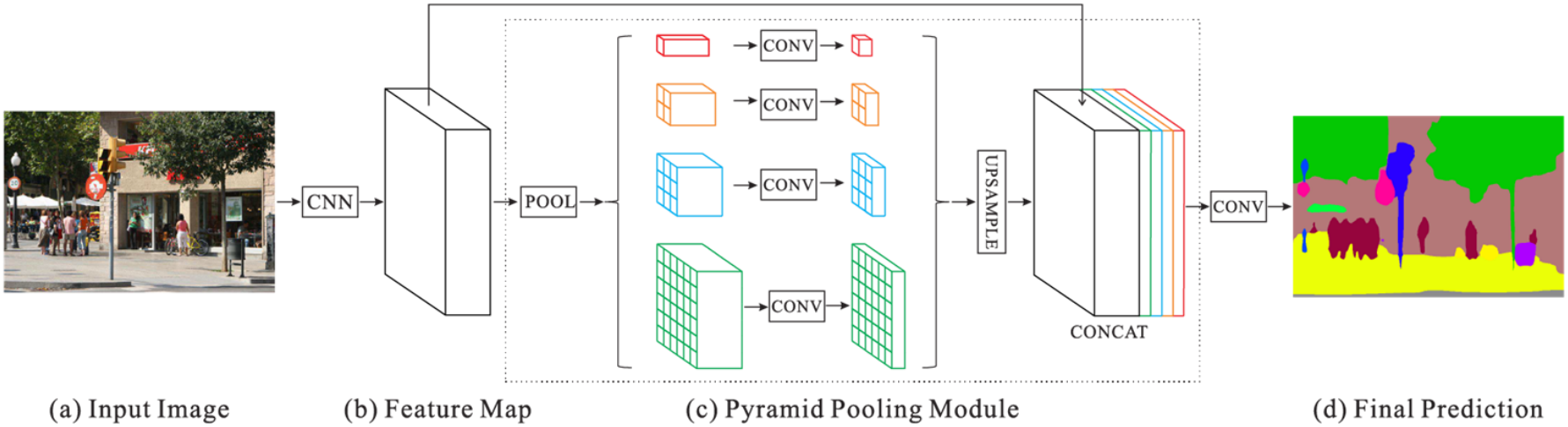


U-net/Segnet



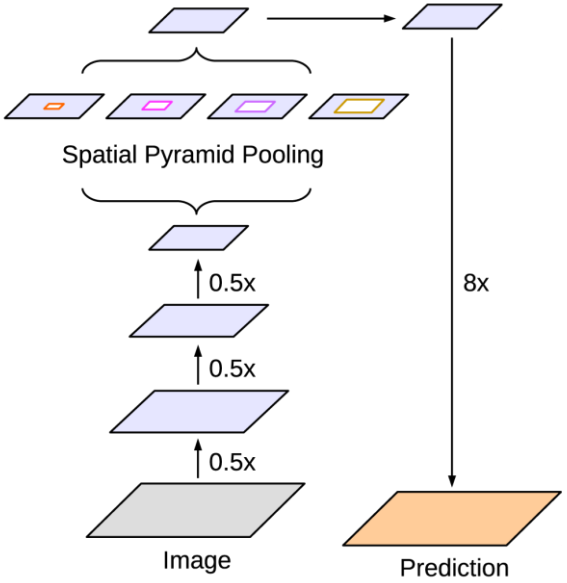
https://arxiv.org/pdf/1511.00561.pdf, https://arxiv.org/pdf/1505.04597.pdf

Spatial pyramid pooling



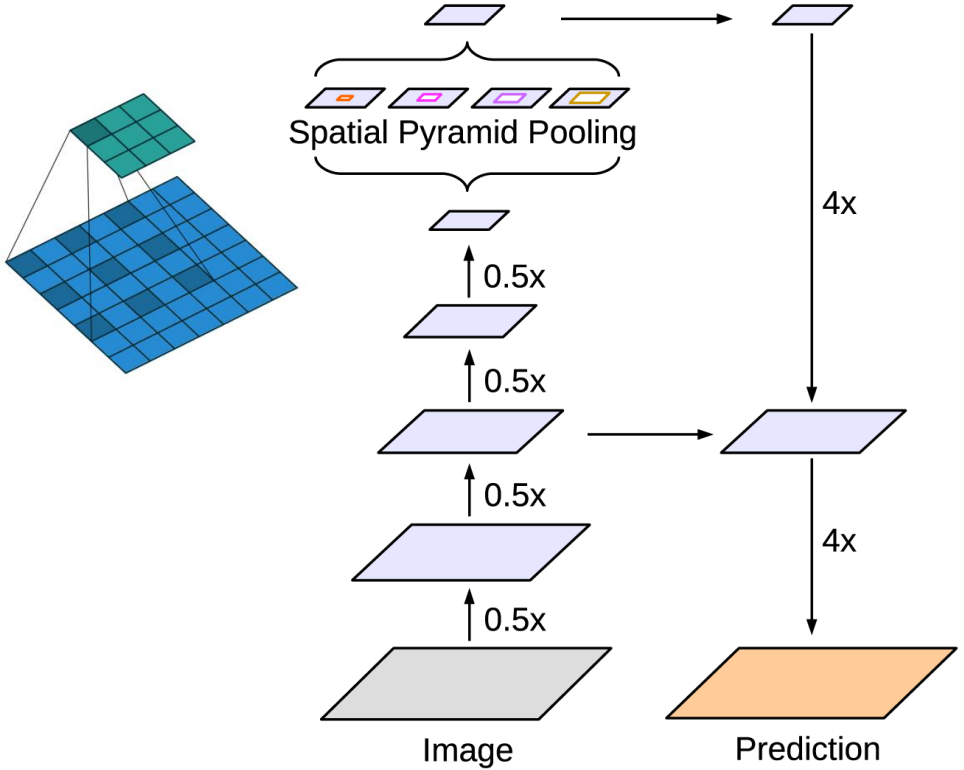
https://arxiv.org/pdf/1612.01105.pdf

Spatial pyramid pooling



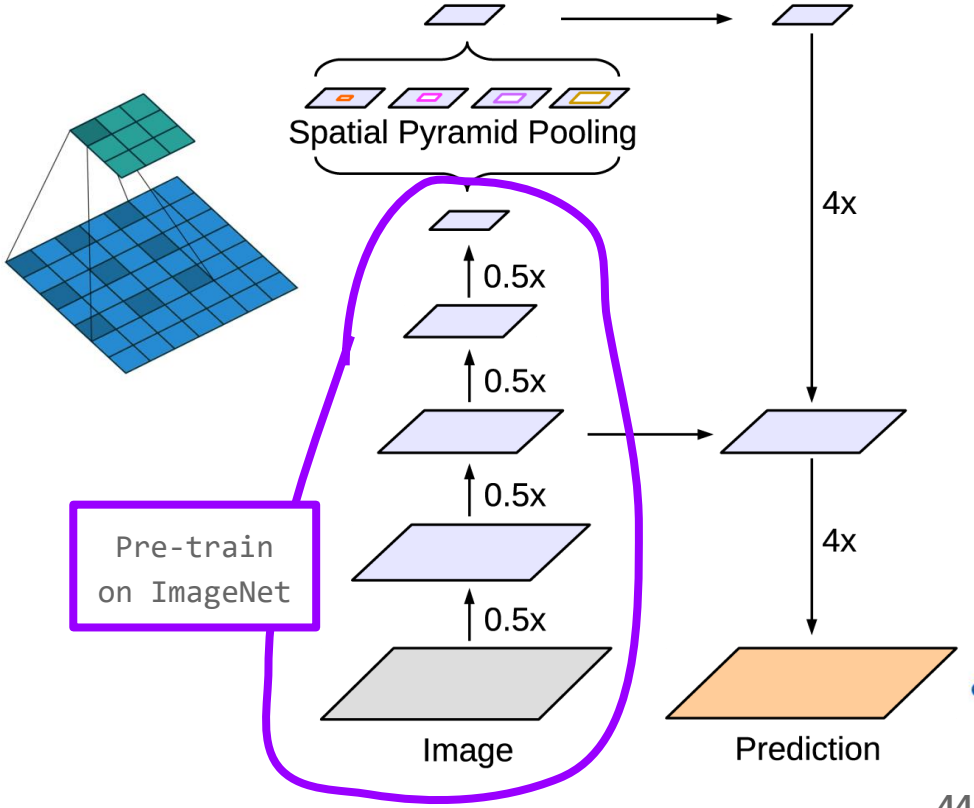
DeepLabv3+

- Atrous convolutions
  - Spaced inputs



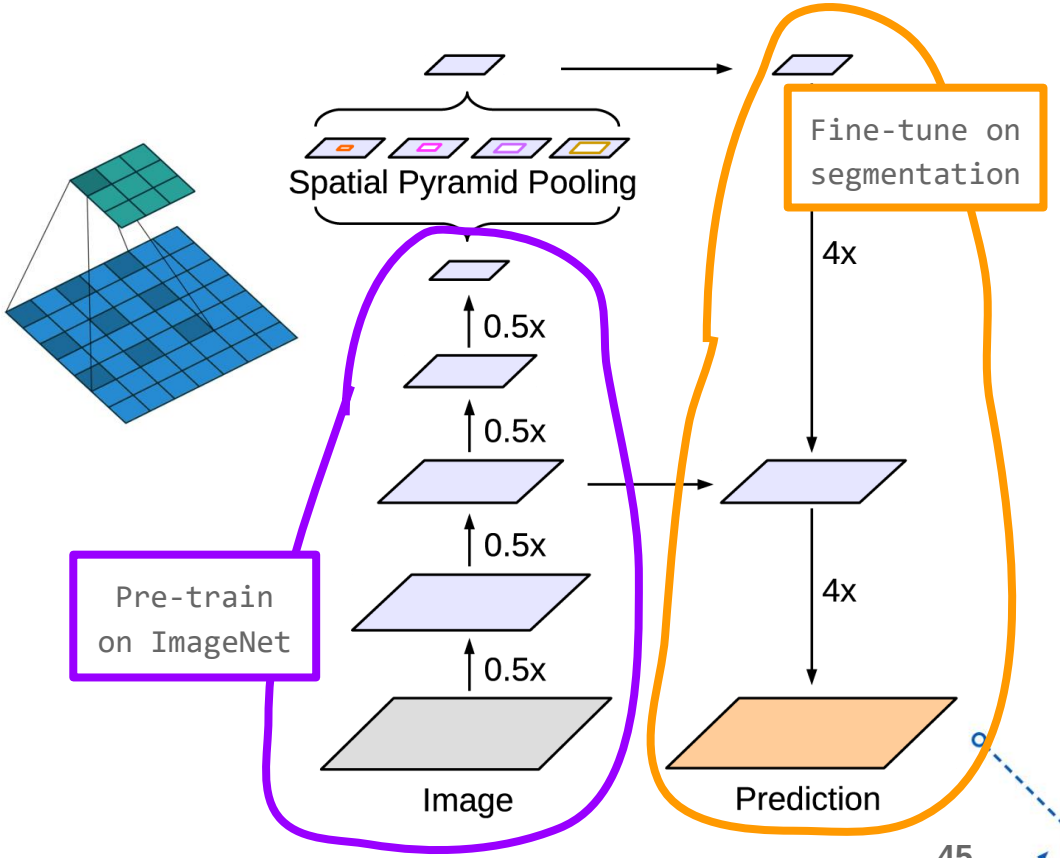
DeepLabv3+

- Atrous convolutions
  - Spaced inputs



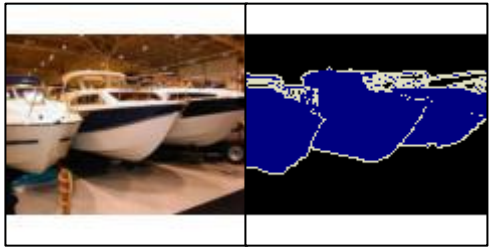
DeepLabv3+

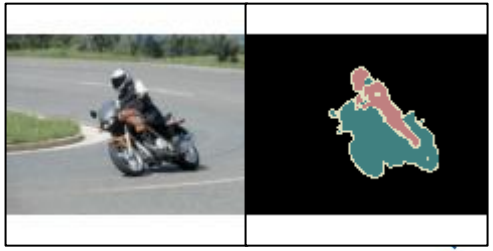
- Atrous convolutions
  - Spaced inputs



PASCAL VOC

- Segmentation dataset:
  - 20 classes
  - 9,993 images with pixel-level labels
- Classes:
  - aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, diningtable, dog, horse motorbike, person, pottedplant, sheep, sofa, train, tvmonitor





PASCAL VOC

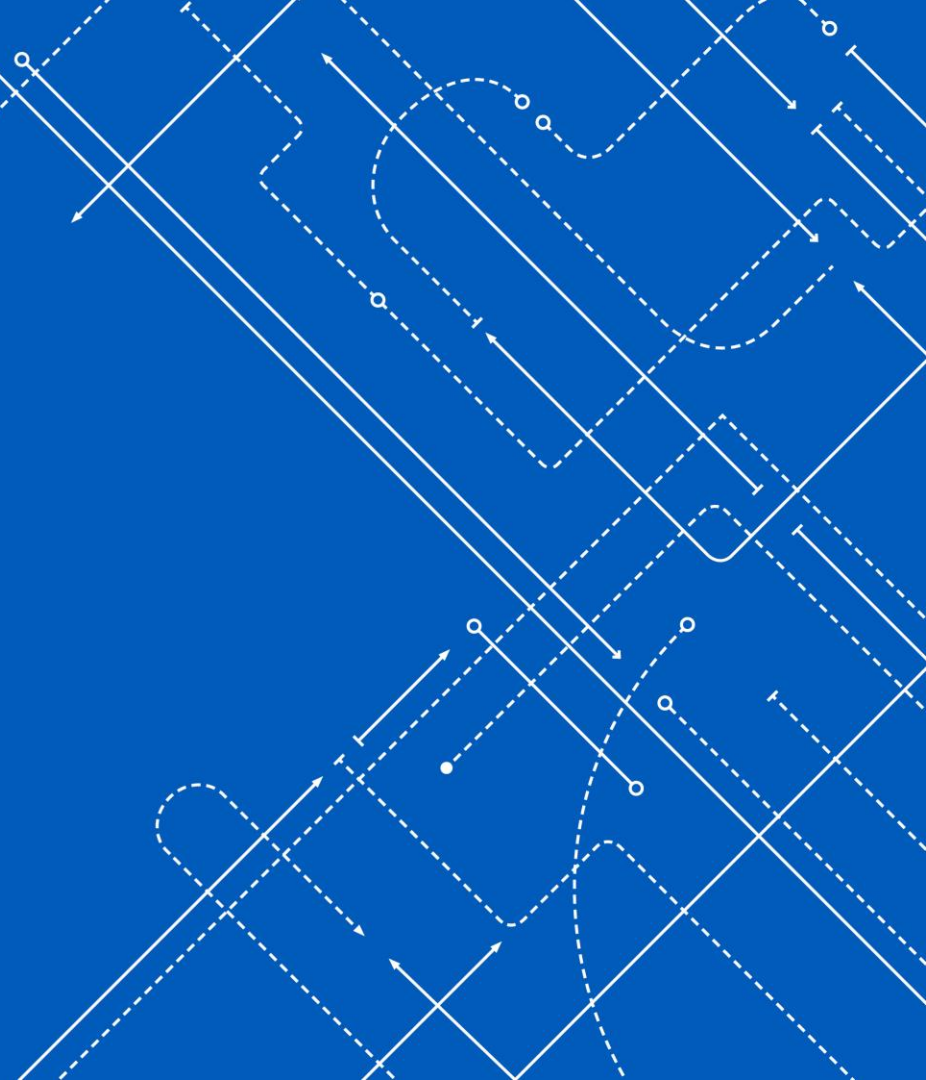
- Segmentation dataset:
  - 30 classes, 50 cities, different seasons, daytime, good weather
  - 5,000 images with fine labels, 20,000 with dense labels
- Classes:
  - road, sidewalk, parking, rail track, building, wall, fence, guard rail, bridge, tunnel, pole, polegroup, traffic light, traffic sign, vegetation, terrain, sky, person, rider, car, truck, bus, caravan, trailer, train, motorcycle, bicycle, license plate

PASCAL VOC

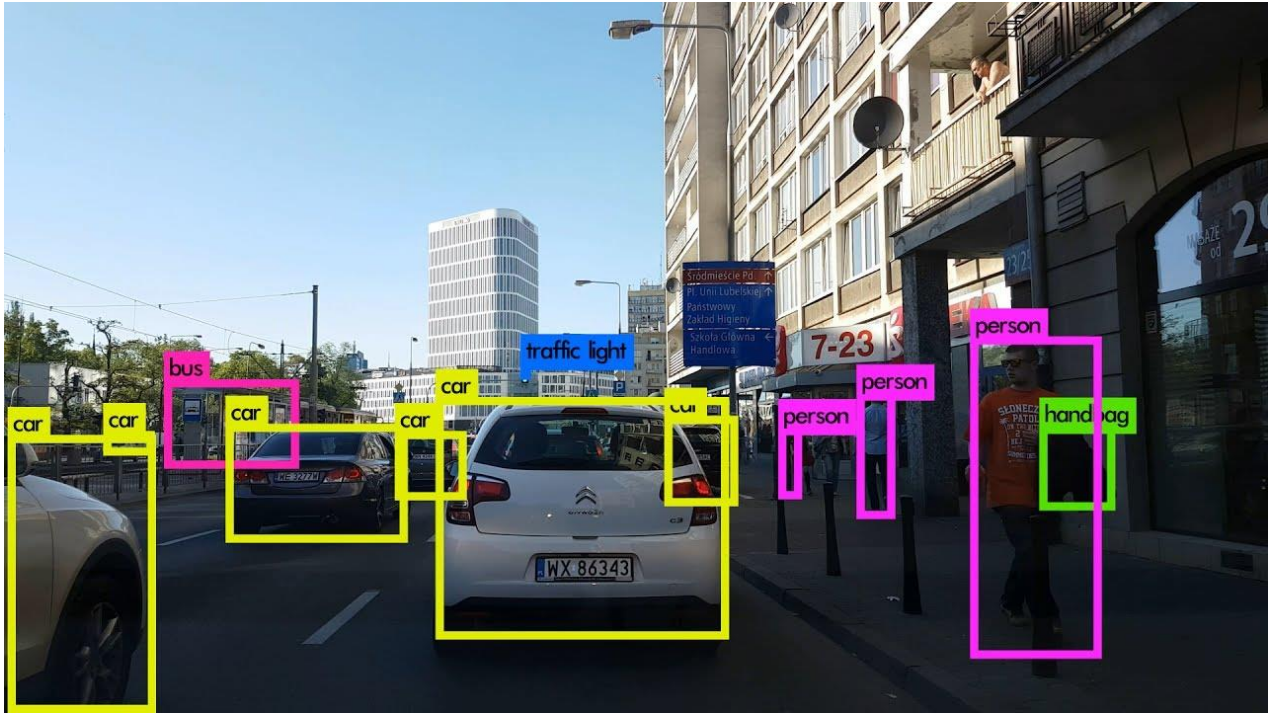
PASCAL VOC

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

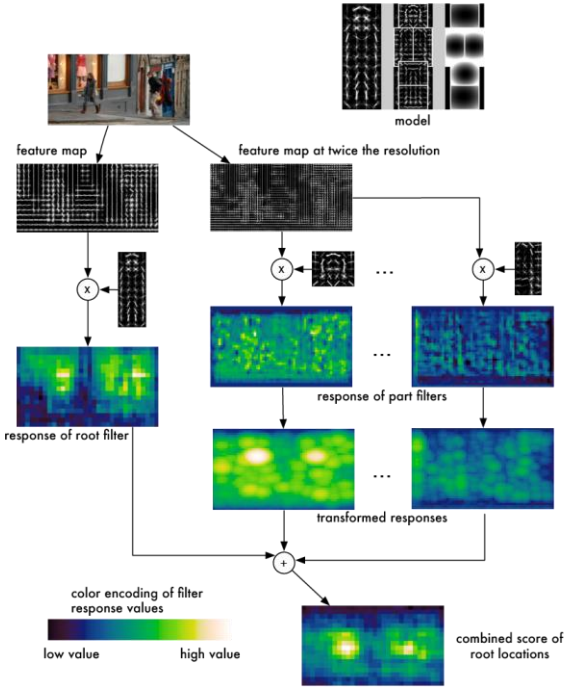
OBJECT DETECTION



Object Detection



Deformable parts models



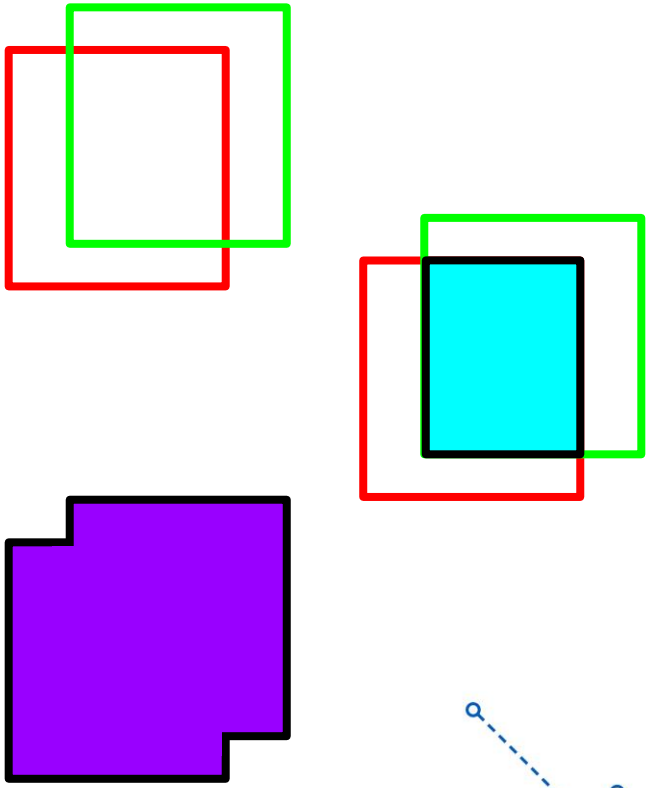
Scoring object detection

- Multiple classes, multiple objects per images
  - Can't just use accuracy
- "Correct" bounding box:
  - Intersection / Union > 0.5
- Intersection:

$$ground\ truth \cap prediction$$

- Union:

$$ground\ truth \cup prediction$$

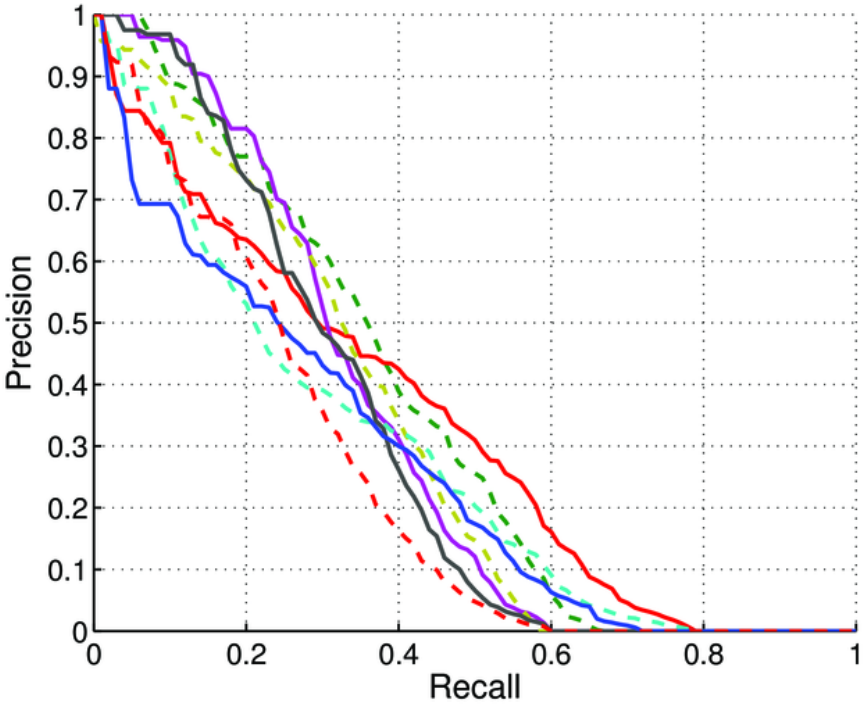


Scoring object detection

- "Correct" bounding box:
  - Intersection / Union > 0.5
- Recall:
  - Correct bounding boxes / total ground-truth boxes
- Precision:
  - Correct bounding boxes / total predicted boxes
- Only the most confident predictions:
  - High precision, low recall
- All the predictions:
  - Low precision, high recall

Scoring object detection

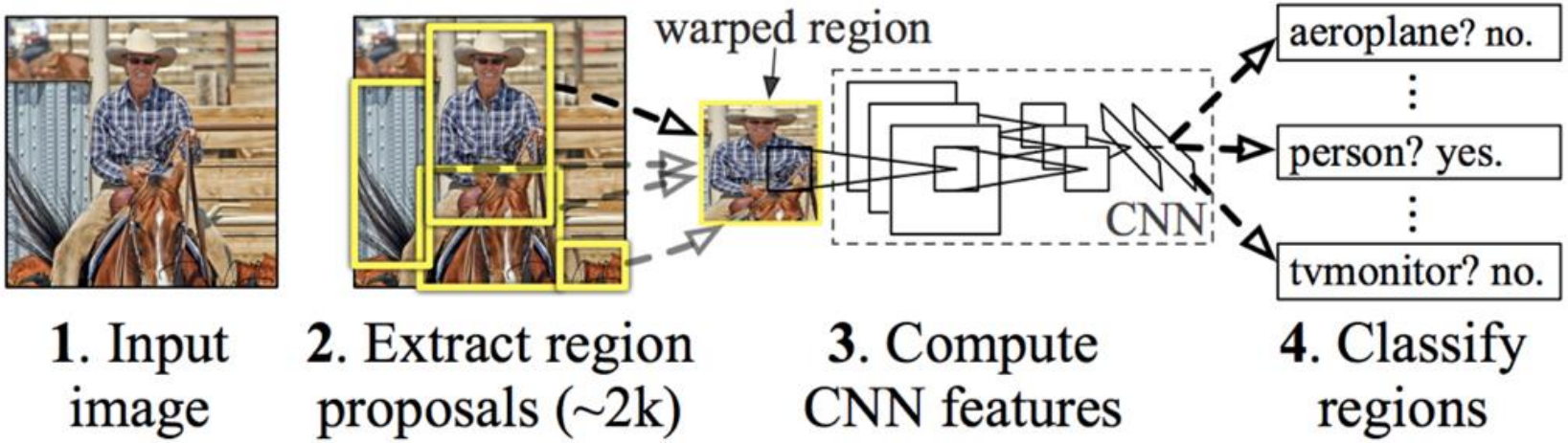
- Precision-Recall curve: vary threshold, plot precision and recall
- Average precision:
  - Area under PR curve
  - Only for a single class
- Take mean of AP across classes:
  - Mean AP (mAP)
  - Standard detection metric
  - Sometimes at particular IOU
    - I.e., mAP@.5 or mAP@.75



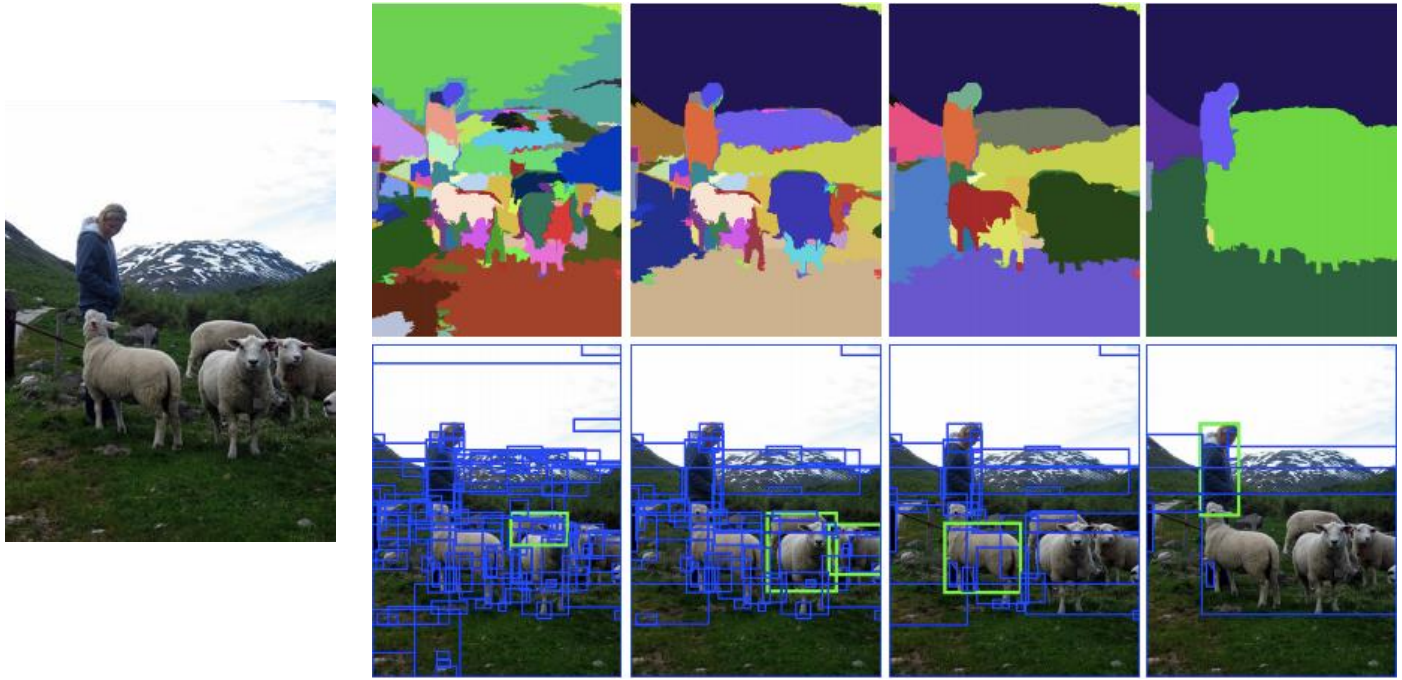
PASCAL VOC

- One of the first large detection datasets:
  - 20 classes
  - 11,530 training images
  - 27,450 annotated objects
- DPM:
  - 33.6% mAP
- DPM is pre-neural network, how do we use CNNs for detection?

R-CNN: Regions with CNN features

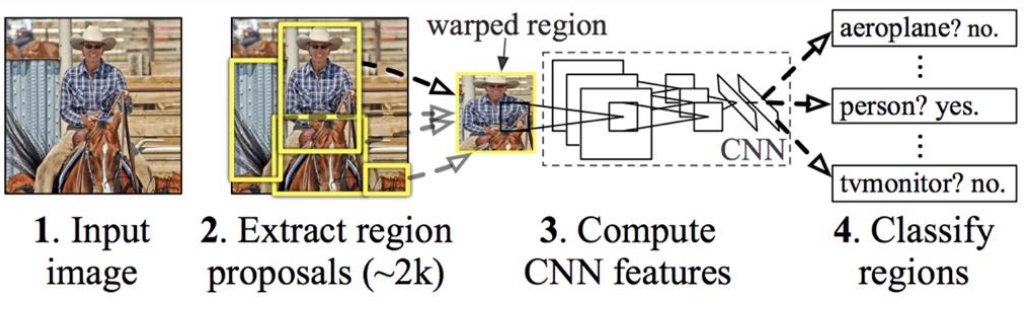


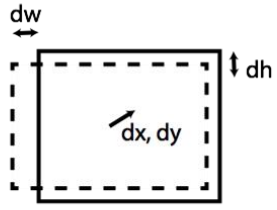
Selective search: fewer proposals

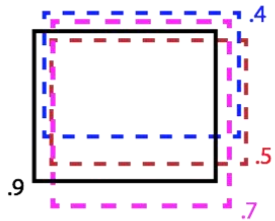


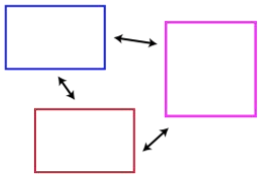
Lots of post processing, 20 sec/image

- Pascal VOC:
  - AlexNet
    - 53.3% mAP
  - VGG-16
    - 62.4% mAP

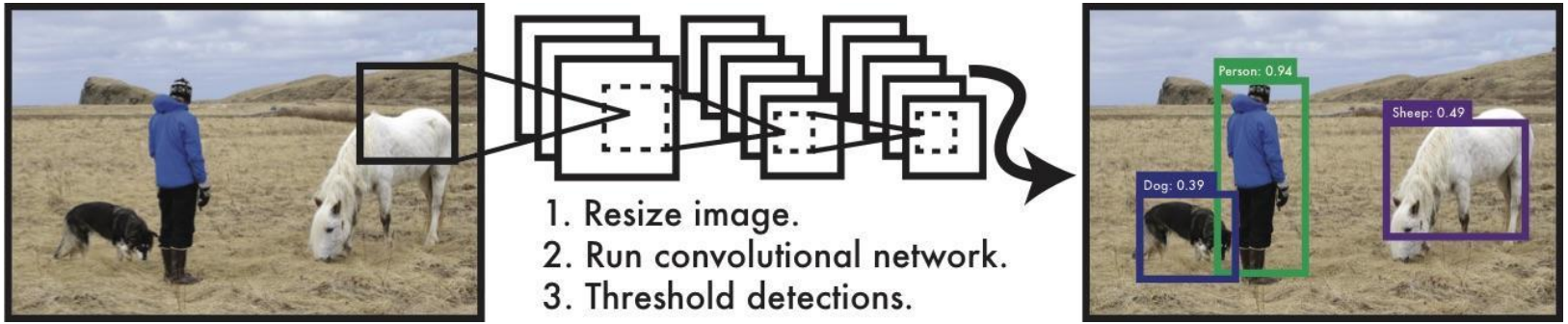








YOLO



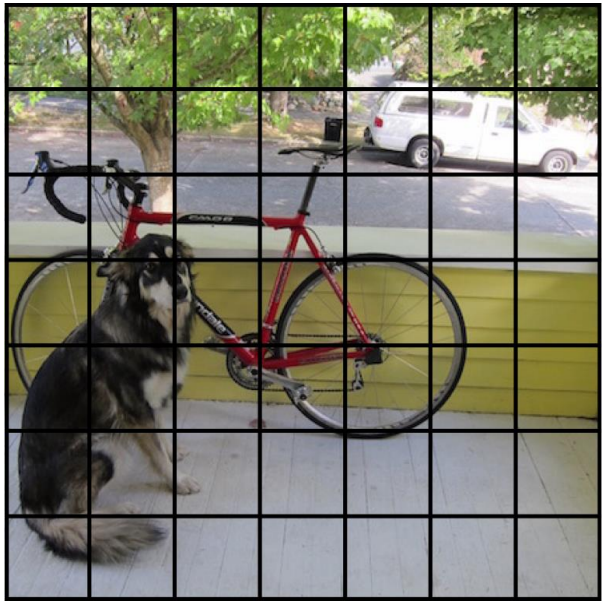
Dr. Joseph Redmon - UW CSE455 - Lec 16

YOLO

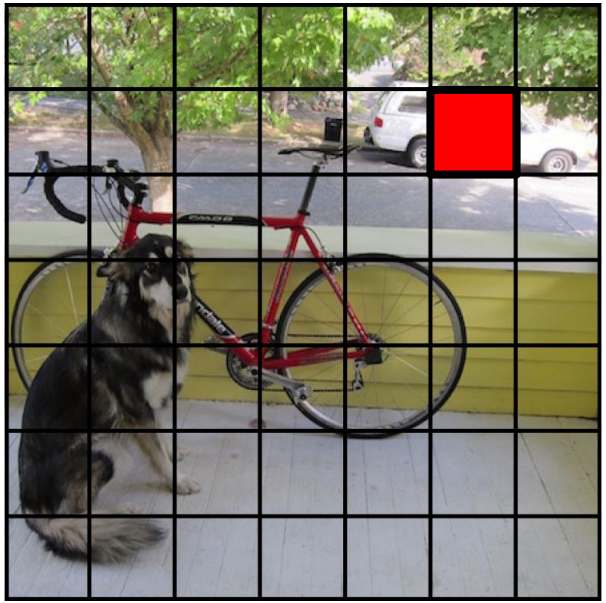
Dr. Joseph Redmon - UW CSE455 - Lec 16

Say you have an image...

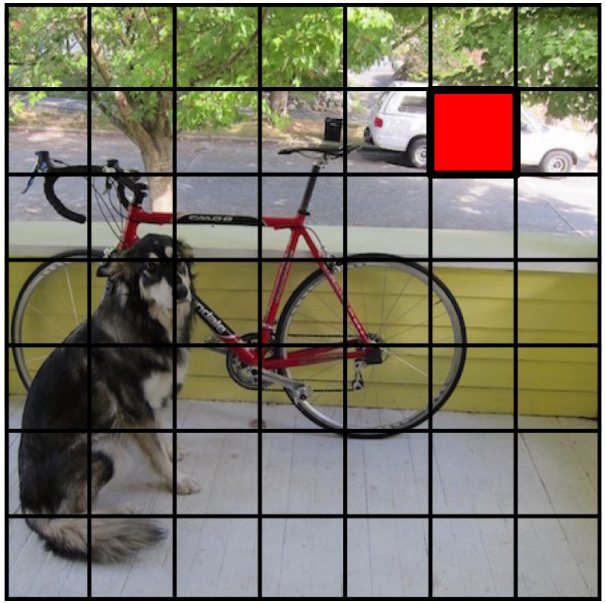
Say you have an image...



For each cell predict $P(obj)$

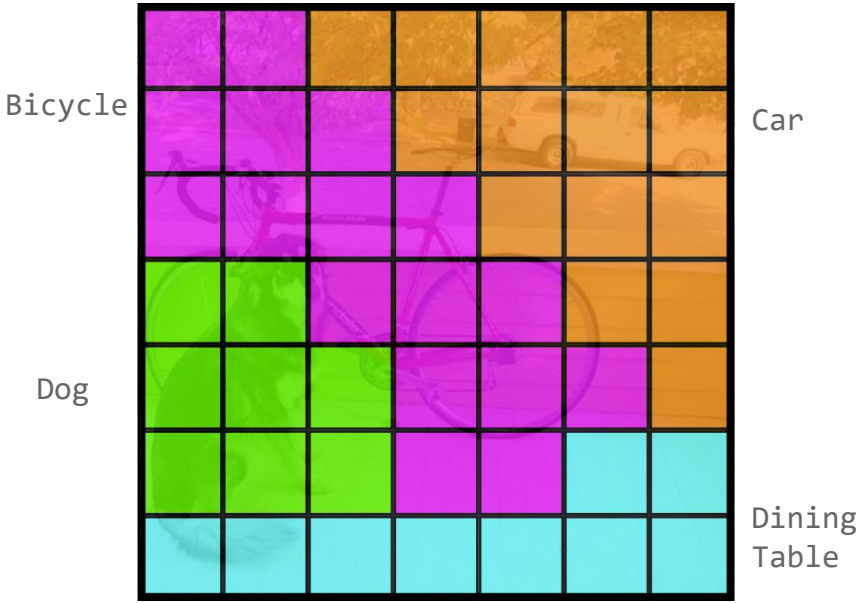


Also predict a bounding box

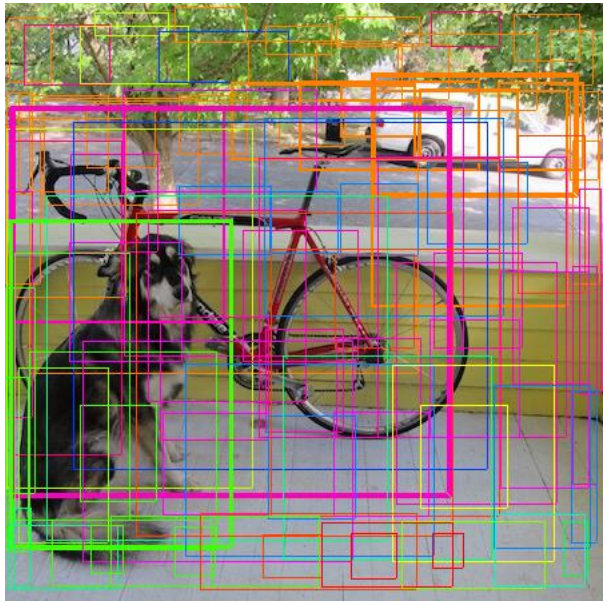


Also predict a bounding box

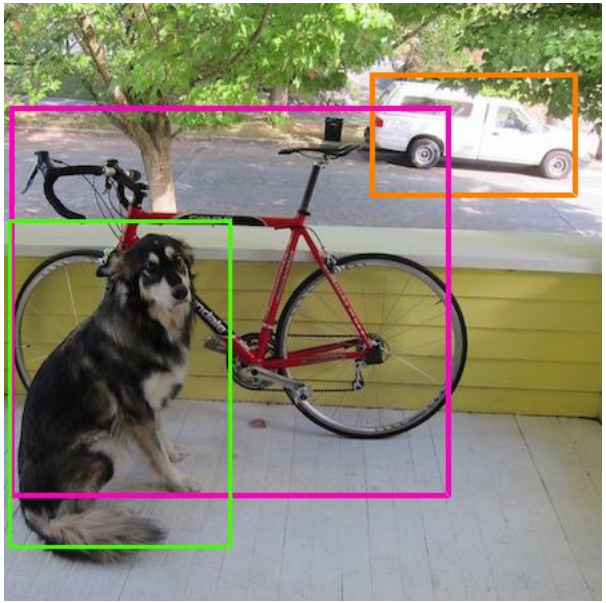
And class probabilities



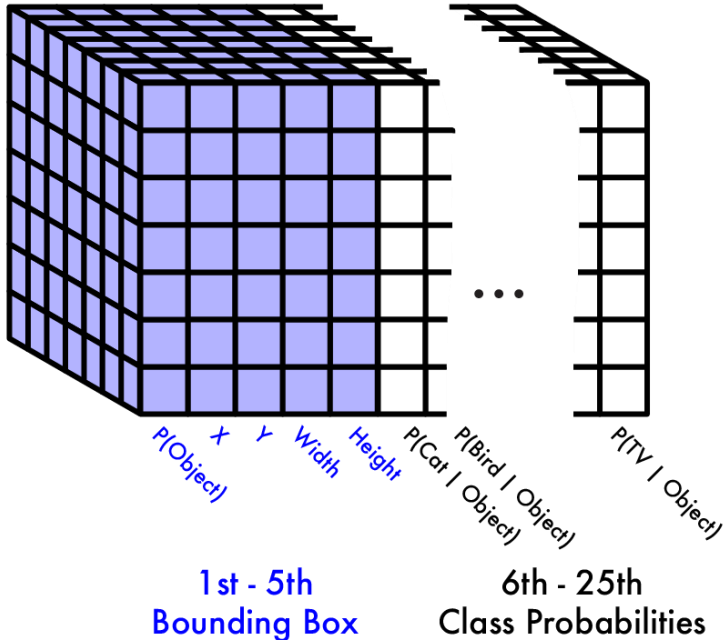
And class probabilities



Threshold and non-max suppression



Tensor encoding detection

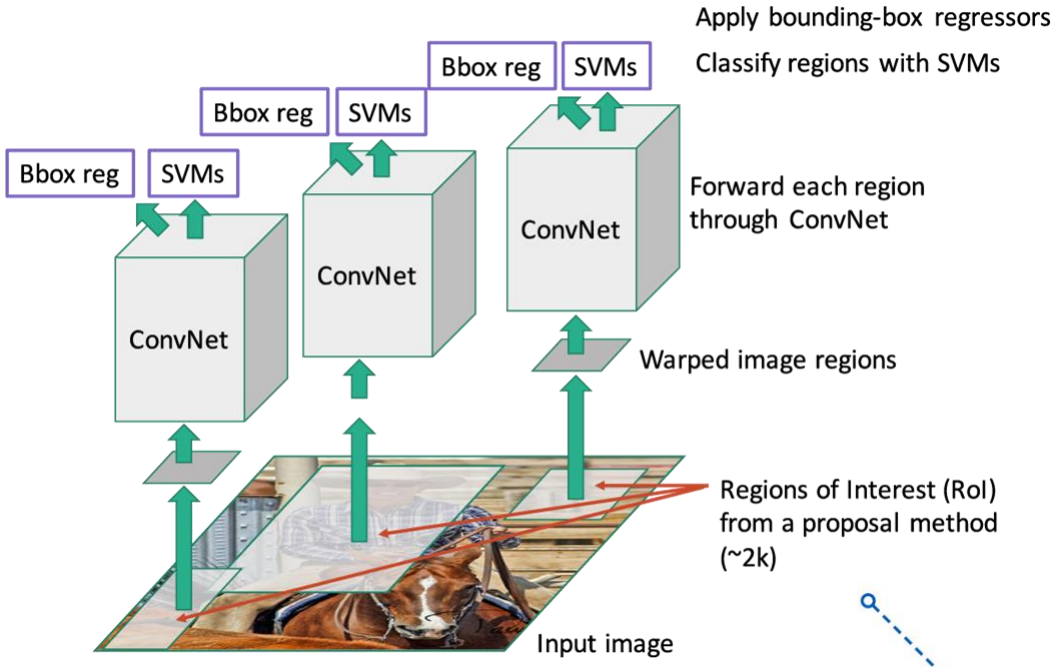


Loss Function

- $L(YOLO) = a_1 L\,(\text{confidence}) + \alpha_2 L(\,\text{localization}\,) + a_3 L\,(\text{classification})$
- Can tune all the alphas
  - $L(\text{confidence})$: binary cross-entropy
    - How sure are we that the box contains an object?
  - $L(\text{localization})$: RMSE
    - How well does our bounding box match ground truth?
  - $L(\text{classification})$: multi-class cross-entropy or $1\ \text{v}\ \text{all}$ binary cross-entropy
    - Did we get the object class right?

R-CNN is slow

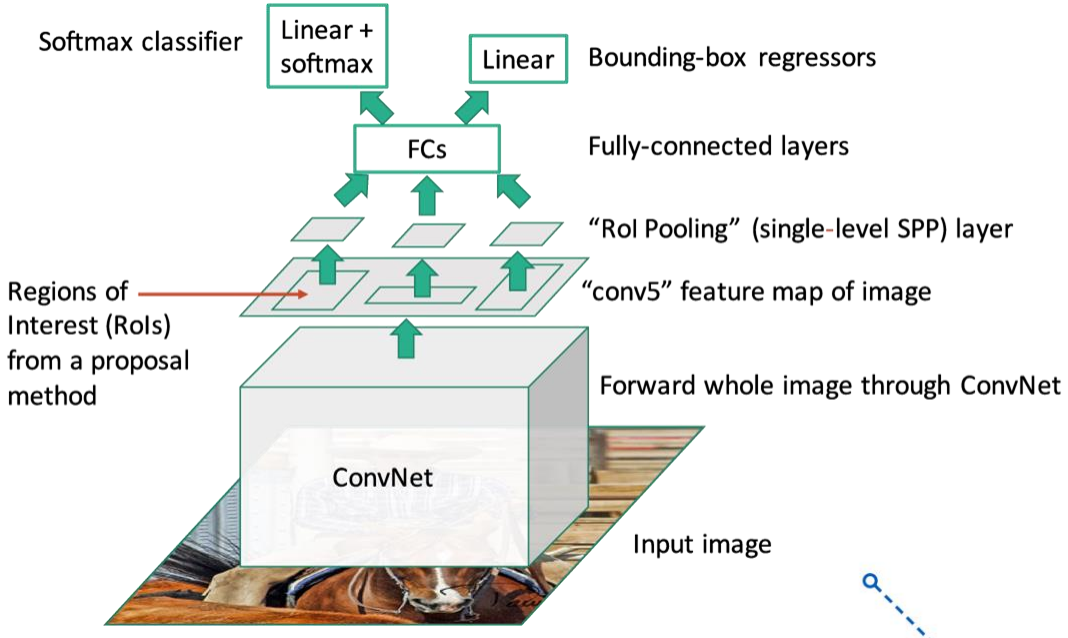
- Run convnet independently over every ROI



https://dl.dropboxusercontent.com/s/vlyrkgd8nz8gy5l/fast-rcnn.pdf?dl=0

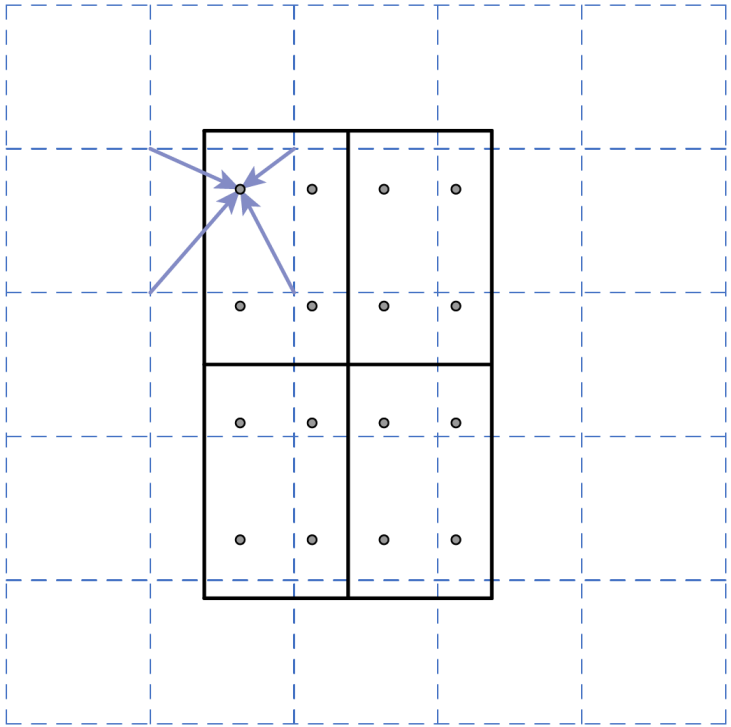
Fast R-CNN

- Run convnet once, extract features using ROI pooling
- ROI Pool:
  - Convert variable sized ROI to fixed size output



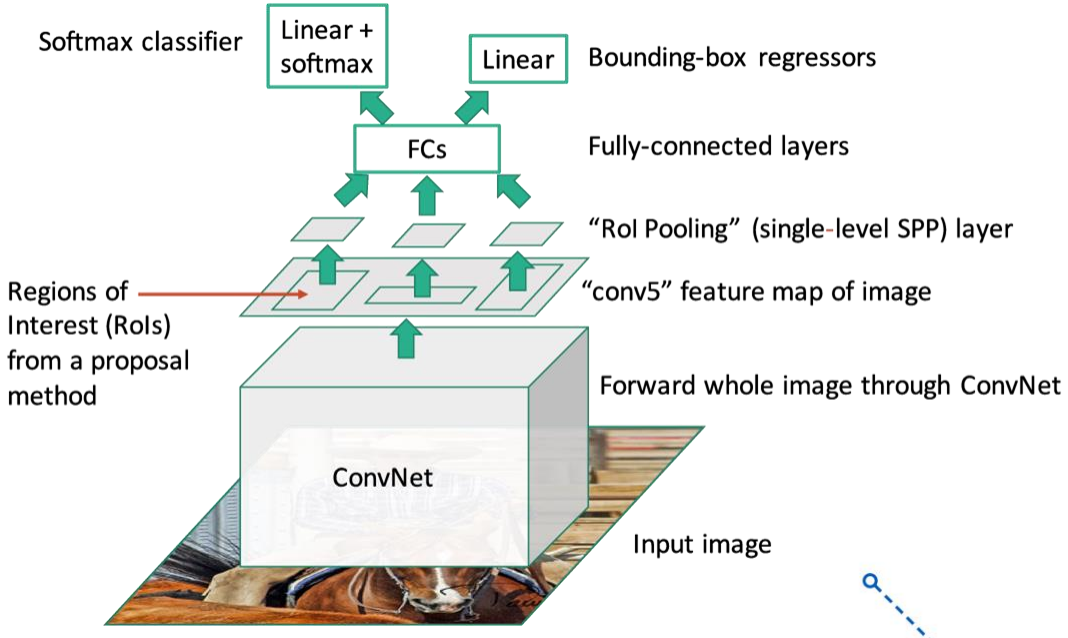
ROI Align

- Better than ROI Pool so we'll talk about it instead
- Split ROI into fixed size (say 2x2)
- Sample image at multiple points for each cell in ROI (bilinear interp.)
- Pool over these samples (max, avg…)



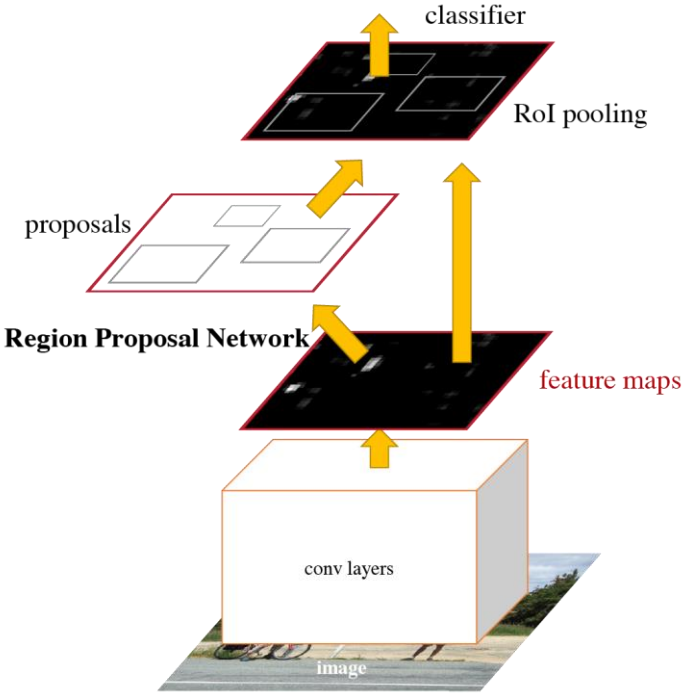
Fast R-CNN

- Run convnet once, extract features using ROI pooling
- ROI Pool:
  - Convert variable sized ROI to fixed size output
- Much faster, no independent network evals (except last linear layer)
- Still slow region proposer, selective search takes ~2 sec



Faster R-CNN

- Use Convnet to propose regions and generate features
- ROI Pool to fix size of ROI features
- Additional layers to classify and predict bbox for ROIs



Saturating PASCAL VOC, need new data

Average Precision (AP %)

	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor	submission date
FOCAL_DRFCN(VOC+COCO, single model) [?]	**88.8**	95.0	93.3	**91.8**	**82.9**	**81.9**	91.6	93.0	97.1	**76.7**	92.5	71.7	96.2	94.9	**94.2**	**93.7**	75.3	**93.3**	80.0	**94.7**	85.4	01-Mar-2018
R4D_faster_rcnn [?]	88.6	94.6	92.3	91.3	82.3	79.4	**91.8**	91.8	97.4	76.6	93.6	**75.3**	97.0	94.6	93.5	92.6	75.1	92.0	80.9	94.4	**86.5**	20-Nov-2016
R-FCN, ResNet Ensemble(VOC+COCO) [?]	88.4	94.8	92.9	90.6	82.4	81.8	89.9	91.7	97.1	76.0	93.4	71.9	96.6	94.3	93.9	92.8	**75.7**	91.9	80.8	93.6	86.4	09-Oct-2016
HIK_FRCN [?]	87.9	95.0	93.2	91.3	80.3	77.7	90.6	89.9	**97.8**	72.8	**93.7**	70.7	**97.2**	**95.4**	94.0	91.8	72.7	92.8	81.1	94.1	86.2	19-Sep-2016
** VIM_SSD ** [?]	87.6	**95.3**	92.0	88.7	81.6	78.5	91.4	**93.2**	95.7	74.9	91.6	73.5	94.2	93.0	93.2	93.0	70.5	93.0	79.1	94.3	85.0	11-May-2018
** Deformable R-FCN, ResNet-101 (VOC+COCO) ** [?]	87.1	94.0	91.7	88.5	79.4	78.0	89.7	90.8	96.9	74.2	93.1	71.3	95.9	94.8	93.2	92.5	71.7	91.8	78.3	93.2	83.3	23-Mar-2017
RefineDet (VOC+COCO,single model,VGG16,one-stage) [?]	86.8	94.7	91.5	88.8	80.4	77.6	90.4	92.3	95.6	72.5	91.6	69.9	93.9	93.5	92.4	92.6	68.8	92.4	78.5	93.6	85.2	16-Mar-2018
FasterRcnn-ResNeXt101(COCO+07++12, single model) [?]	86.8	93.9	**93.4**	88.3	80.2	72.6	89.4	89.3	96.8	73.0	91.5	72.3	95.4	94.5	93.8	91.7	70.7	90.6	**81.2**	92.6	83.9	04-May-2017
** PSSNet(VOC+COCO) ** [?]	85.5	92.4	91.4	85.9	78.6	75.8	88.0	89.8	95.2	72.4	87.8	72.2	94.0	92.7	93.2	92.3	70.7	88.8	76.1	92.1	81.2	30-Mar-2018
R-FCN, ResNet (VOC+COCO) [?]	85.0	92.3	89.9	86.7	74.7	75.2	86.7	89.0	95.8	70.2	90.4	66.5	95.0	93.2	92.1	91.1	71.0	89.7	76.0	92.0	83.4	09-Oct-2016
** MONet(VOC+COCO) ** [?]	84.3	92.4	90.5	84.7	75.4	71.6	87.2	88.9	94.6	70.5	86.9	71.0	92.3	91.8	90.8	91.7	69.8	89.1	75.1	91.3	79.6	01-Apr-2018
PVANet+ [?]	84.2	93.5	89.8	84.1	75.6	69.7	88.2	87.9	93.4	70.0	87.7	**75.3**	92.9	90.5	90.9	90.2	67.3	86.4	80.3	92.0	78.8	26-Oct-2016
FSSD512 [?]	84.2	92.8	90.0	86.2	75.9	67.7	88.9	89.0	95.0	68.8	90.9	68.7	92.8	92.1	91.4	90.2	63.1	90.1	76.9	91.5	82.7	07-Nov-2017
BlitzNet512 [?]	83.8	93.1	89.4	84.7	75.5	65.0	86.6	87.4	94.5	69.9	88.8	71.7	92.5	91.6	91.1	88.9	61.2	90.4	79.2	91.8	83.0	19-Jul-2017
PFPNet512 VGG16 07++12+COCO [?]	83.8	93.0	89.9	85.1	75.8	66.4	88.4	88.3	94.0	67.9	89.5	69.7	92.0	91.8	91.6	88.7	61.1	89.1	78.4	90.5	84.3	18-Oct-2017
Faster RCNN, ResNet (VOC+COCO) [?]	83.8	92.1	88.4	84.8	75.9	71.4	86.3	87.8	94.2	66.8	89.4	69.2	93.9	91.9	90.9	89.6	67.9	88.2	76.8	90.3	80.0	10-Dec-2015

Common Objects in COntext (COCO)

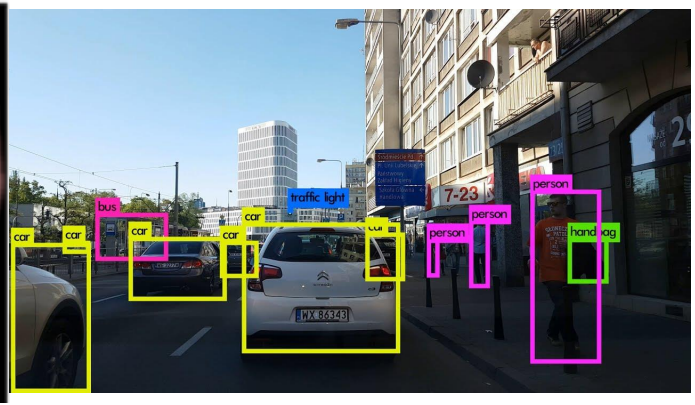
- 80 objects
- 117,261 train/val images
- 902,435 object instances
- New detection metric, mAP averaged over IOU $[.5 - .95]$
- Segmentation masks for each instance
- *Originally by Microsoft*



Segmentation vs Detection

- Pixel-level labels
- Category only

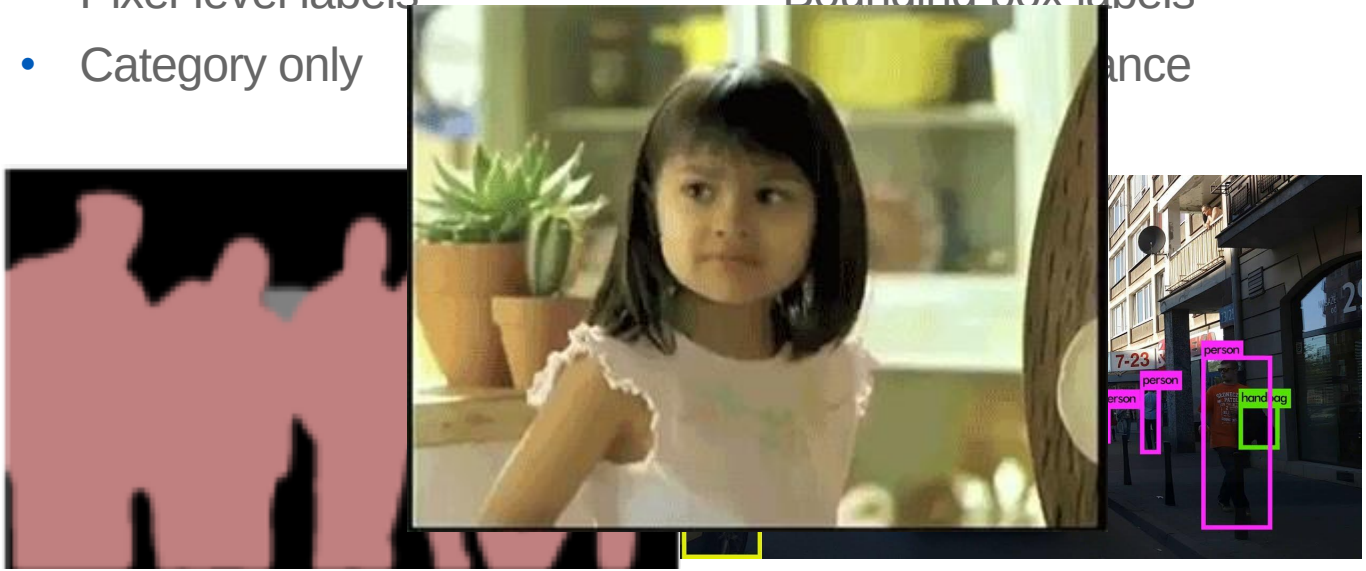
- Bounding box labels
- Category + instance



Segmentation vs Detection

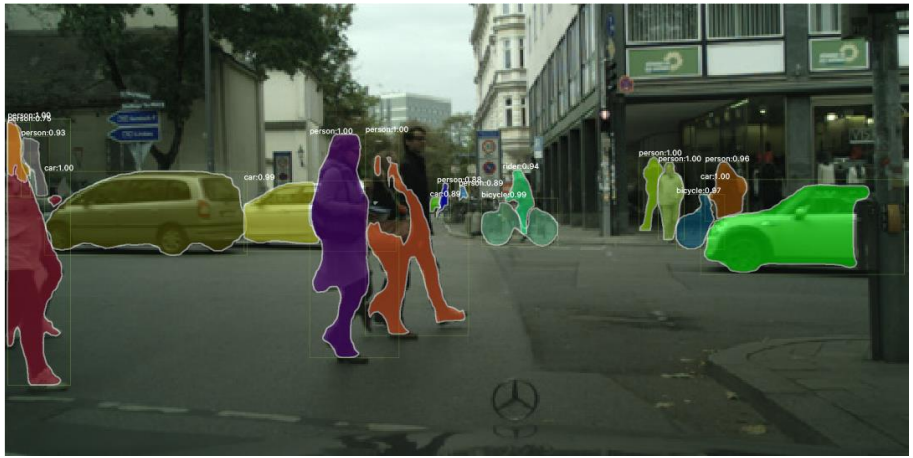
- Pixel-level labels
- Category only

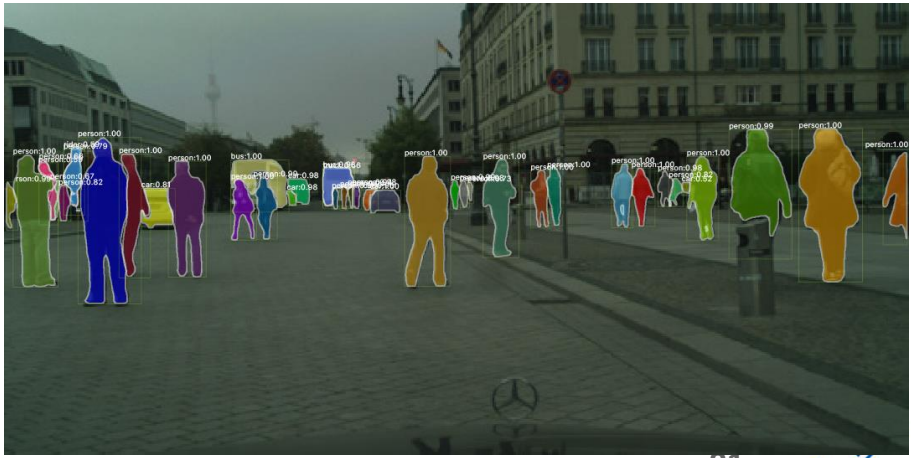
- Bounding box labels
- ance



Instance Segmentation

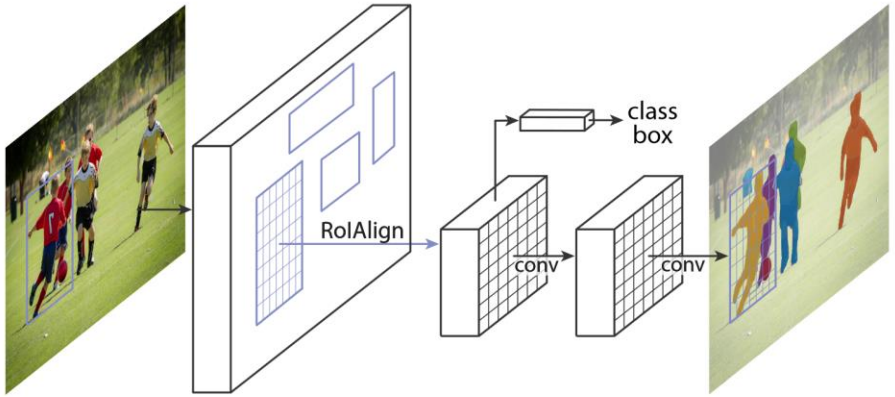
- Given an image produce instance-level segmentation
- Which class does each pixel belong to
- Also which instance





Mask R-CNN

- Similar to R-CNN but predict mask as well as box



Mask R-CNN

- Similar to R-CNN but predict mask as well as box