ROB 498/599: Deep Learning for Robot Perception (DeepRob)

Lecture 11: Object Detection - part 1

02/17/2025

https://deeprob.org/w25/

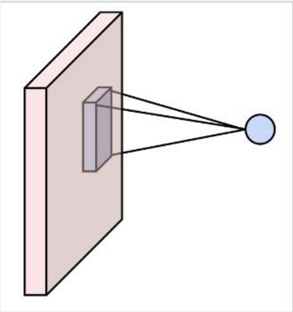
ROBOTICS

Today

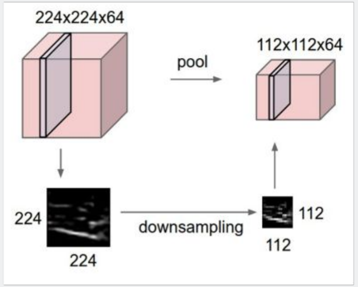
- Recap (5min)
- Object Detection
  - Object Detection Overview (15min)
  - Region Proposals (10min)
  - R-CNN (25min)
    - NMS and mAPs
  - Fast R-CNN (15min)
- Summary and Takeaways (5min)

Recap: Components of Convolutional Networks

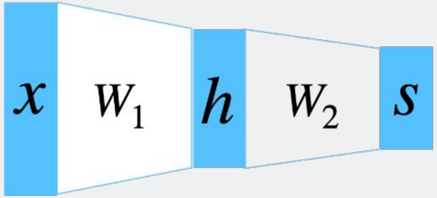
Convolution Layers



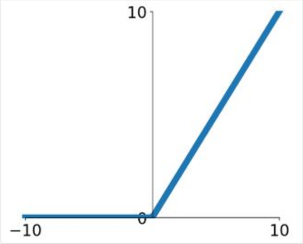
Pooling Layers



Fully-Connected Layers



Activation Function



Normalization

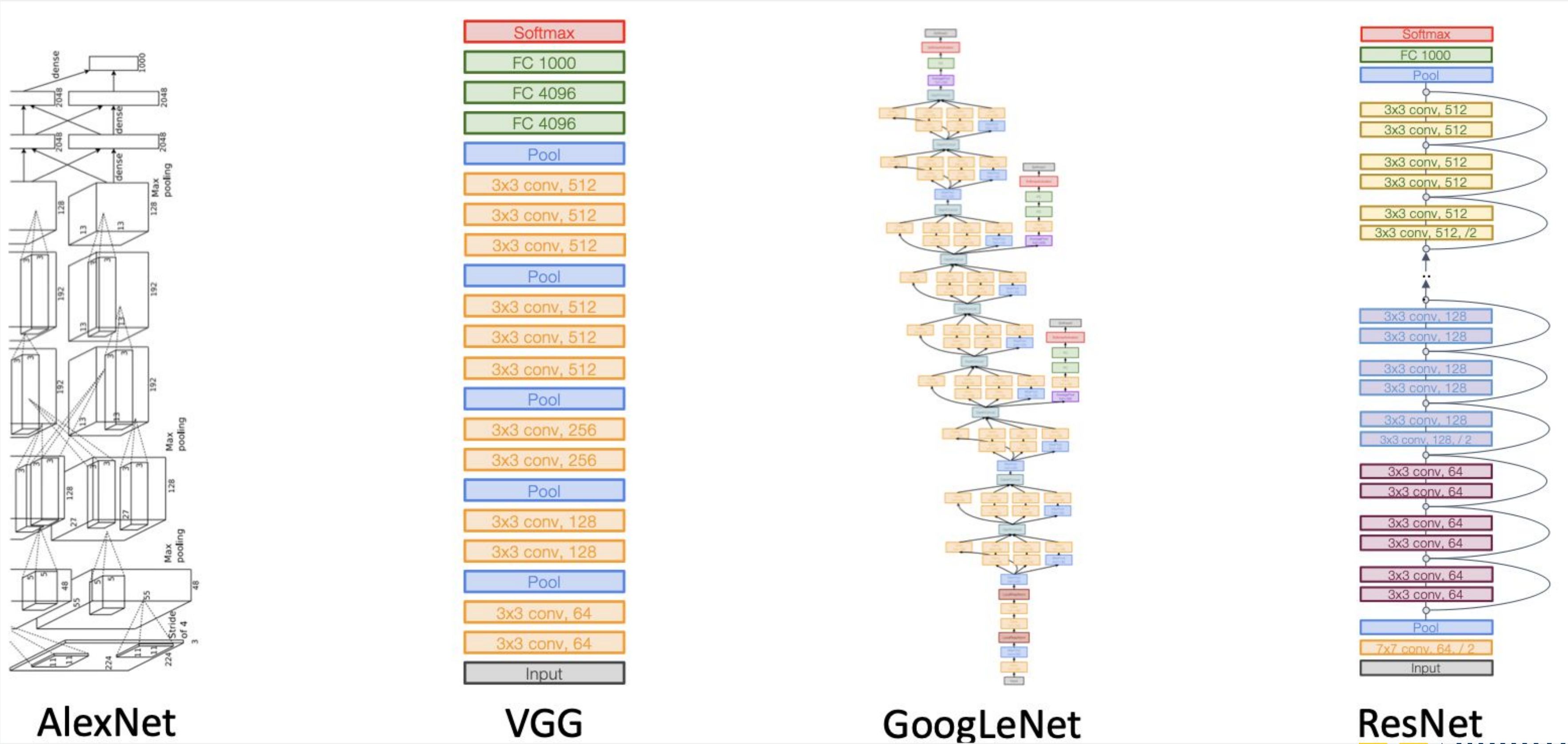
$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \epsilon}}$$

Next Up

Question: How should we put them together?

M | ROBOTICS

Recap: CNN Architectures



Recap: Training NNs

1. One time setup:
   - Activation functions, data preprocessing, weight initialization, regularization
2. Training dynamics:
   - Learning rate schedules; large-batch training; hyperparameter optimization
3. After training:
   - Model ensembles, transfer learning

Upcoming

*note: may have updates - will announce

- **P3 released. P3 Due March 9, 2025 (recommend finish before spring vacation)**
- **Midterm: March 12, 2025, 12pm-1:30pm EST (Wednesday in class after spring vacation)**
  - Pen/Pencil and paper exam
  - 1 A4/Letter-size note, front and back
  - No GenAI/phone/computer/internet
- **Final Project ideas:** https://deeprob.org/w25/projects/finalproject/

So far… Image Classification

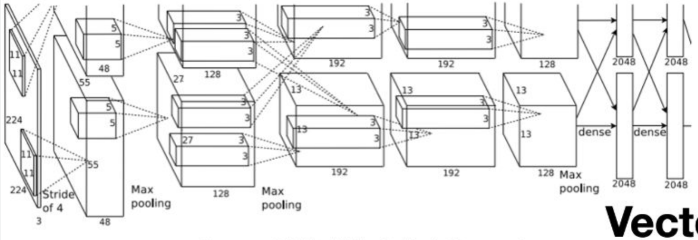


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector:
4096

Fully connected:
4096 to 10

Chocolate Pretzels

Granola Bar

Potato Chips

Water Bottle

Popcorn

Computer Vision Tasks

Classification

"Chocolate Pretzels"

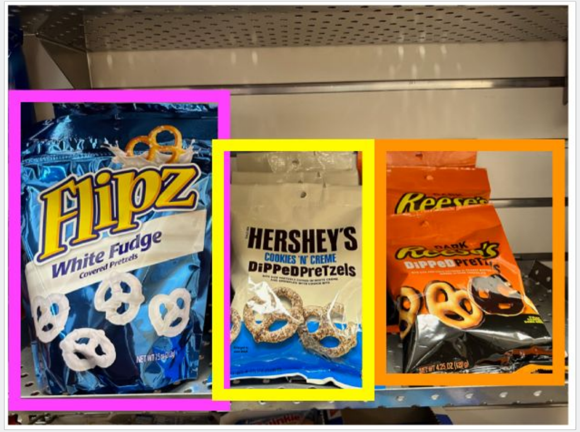
No spatial extent

Semantic Segmentation

Chocolate Pretzels,
Shelf

No objects, just pixels

Object Detection



Instance Segmentation

Flipz, Hershey's, Keese's

Multiple objects

Computer Vision Tasks

Classification

"Chocolate Pretzels"

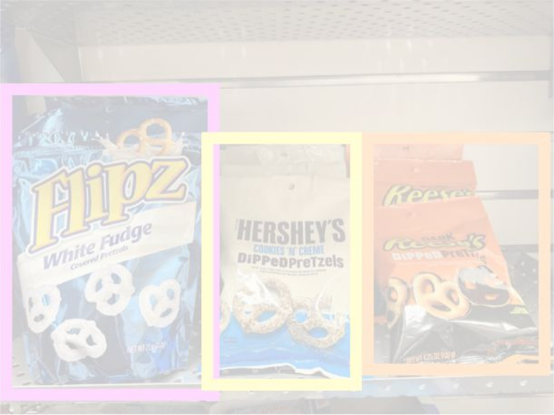
No spatial extent

Semantic Segmentation

Chocolate Pretzels, Shelf

No objects, just pixels

Object Detection



Instance Segmentation

Flipz, Hershey's, Keese's

Multiple objects

Classification: Transferring to New Tasks

Classification

"Chocolate Pretzels"

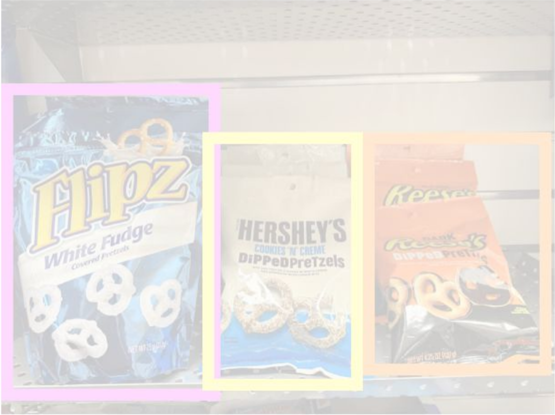
No spatial extent

Semantic Segmentation

Chocolate Pretzels, Shelf

No objects, just pixels

Object Detection



Instance Segmentation

Flipz, Hershey's, Keese's

Multiple objects

Object Detection

Classification

"Chocolate Pretzels"

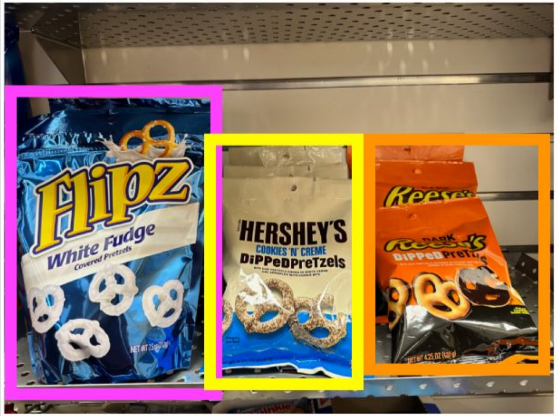
No spatial extent

Semantic Segmentation

Chocolate Pretzels, Shelf

No objects, just pixels

Object Detection



Flipz, Hershey's, Keese's

Multiple objects

Instance Segmentation

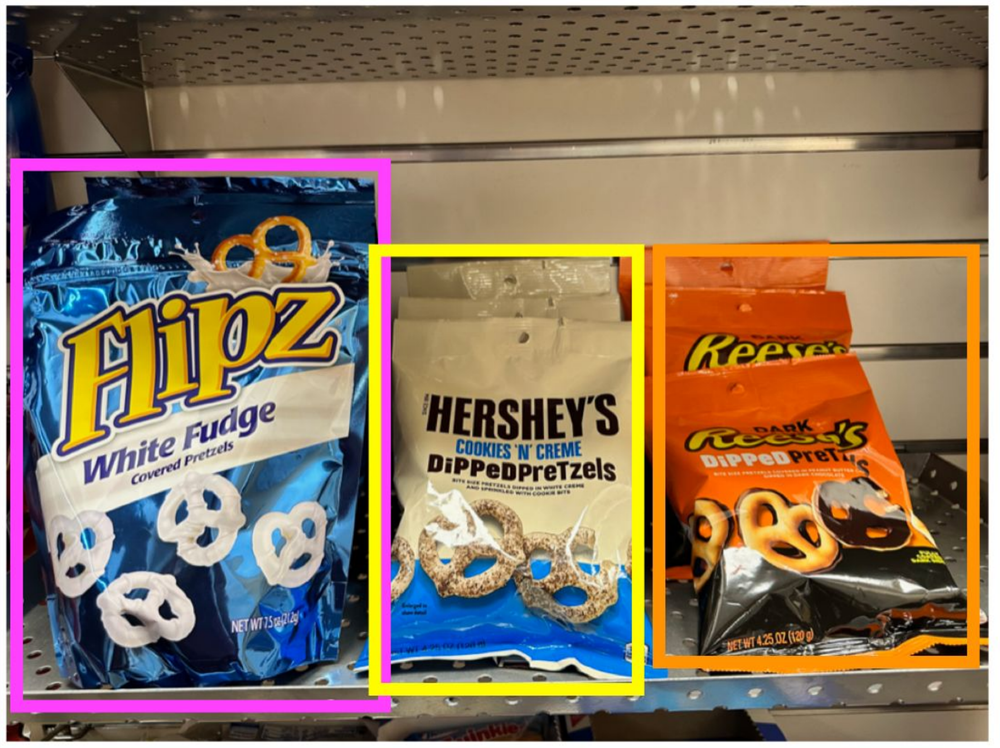
ROBOTICS

Object Detection: Task Definition

Input: Single RGB image

Output: A set of detected objects;
For each object predict:

1. Category label (from a fixed set of labels)
2. Bounding box (four numbers: x, y, width, height)

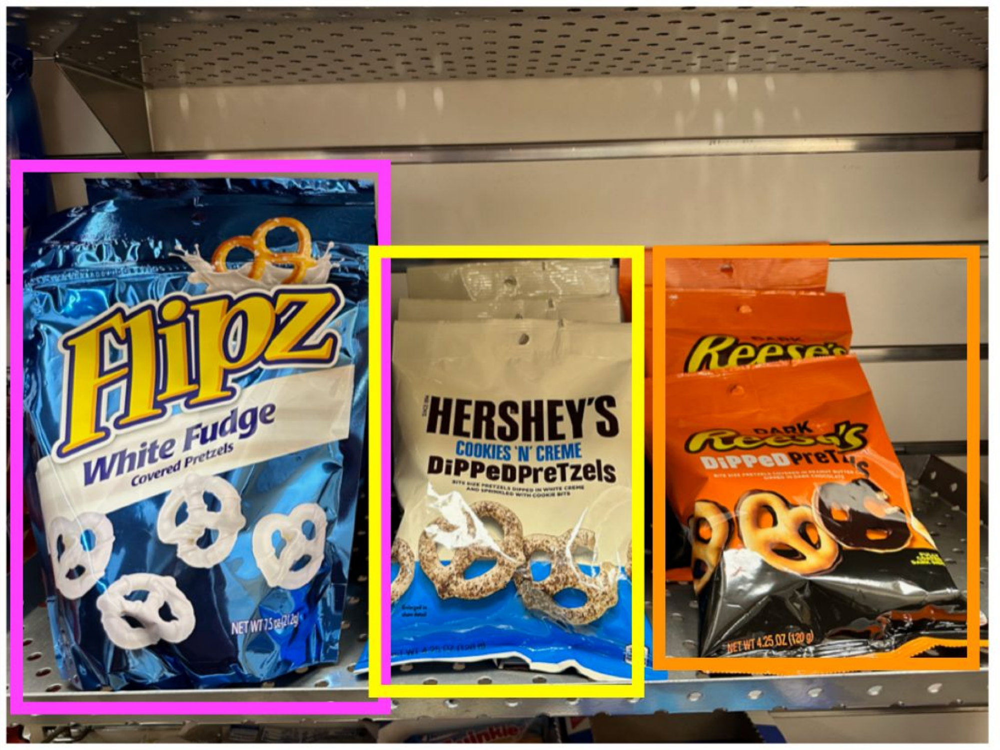


Object Detection: Challenges

Multiple outputs: Need to output variable numbers of objects per image

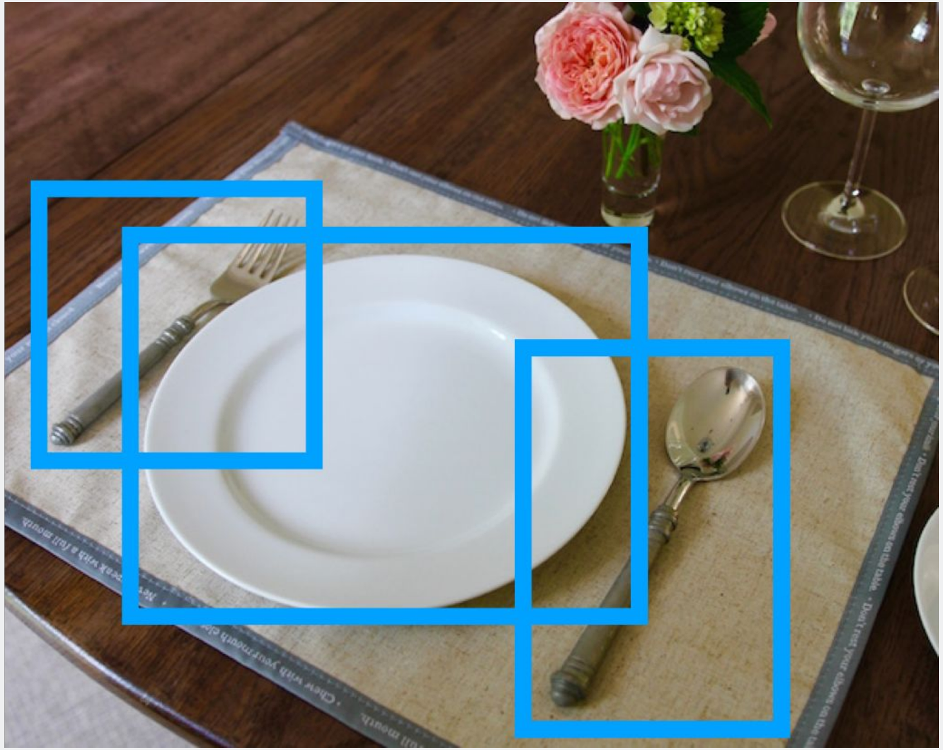
Multiple types of output: Need to predict ”what” (category label) as well as “where” (bounding box)

Large images: Classification works at 224x224; need higher resolution for detection, often ~800x600



Object Detection: Bounding Boxes

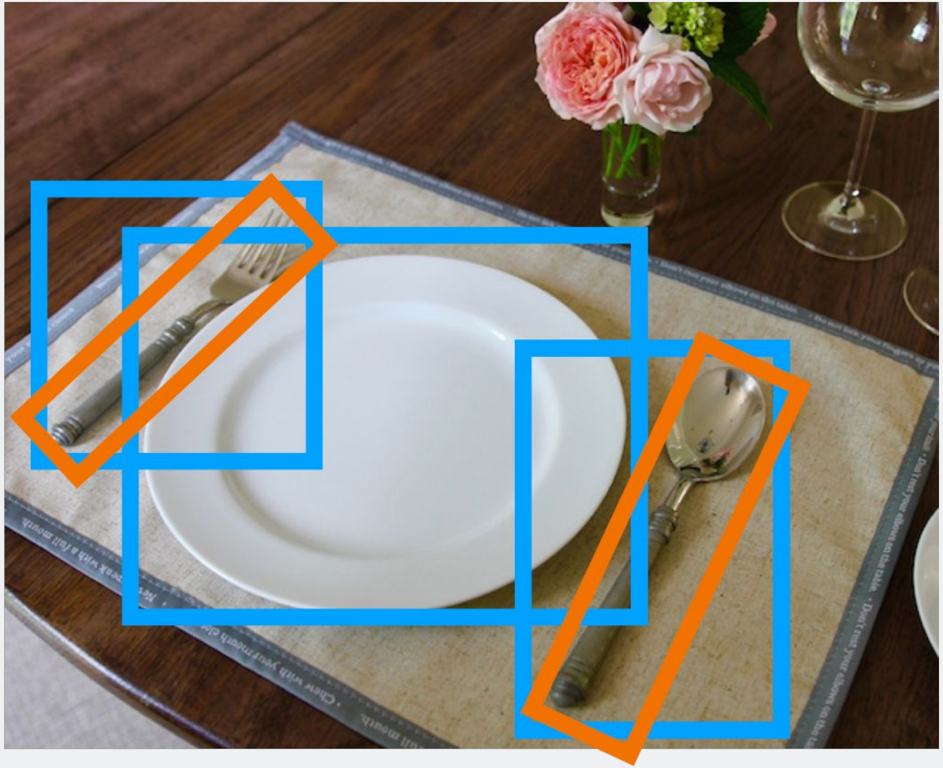
Bounding boxes are typically *axis-aligned*



Object Detection

Bounding boxes are typically *axis-aligned*

Oriented boxes are much less common



ROBOTICS

Object Detection: Modal vs. Amodal Boxes

Bounding boxes cover only the visible portion of the object

Zhu et al, "Semantic Amodal Segmentation", CVPR 2017
https://openaccess.thecvf.com/content_cvpr_2017/papers/Zhu_Semantic_Amodal_Segmentation_CVPR_2017_paper.pdf

Object Detection: Modal vs. Amodal Boxes

Bounding boxes cover only the visible portion of the object

<u>Amodal detection:</u> box covers the entire extent of the object, even occluded parts

Object Detection: Modal vs. Amodal Boxes

"Modal" detection: Bounding boxes (usually) cover only the visible portion of the object

Amodal detection: box covers the entire extent of the object, even occluded parts

Comparing Boxes: IoU (Intersection over Union)

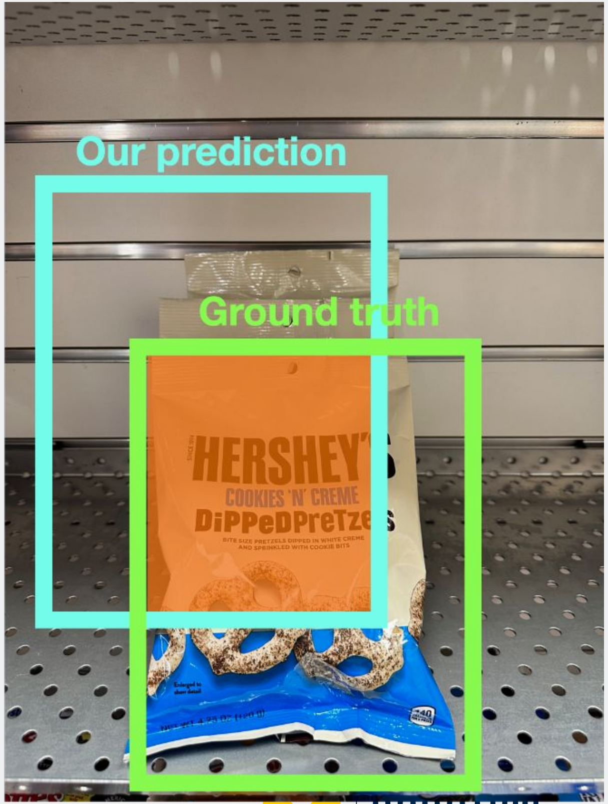
How can we compare our prediction to the ground-truth box?

Comparing Boxes: IoU (Intersection over Union)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU) (Also called "Jaccard similarity" or "Jaccard index"):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$



Comparing Boxes: IoU (Intersection over Union)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU) (Also called "Jaccard similarity" or "Jaccard index"):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

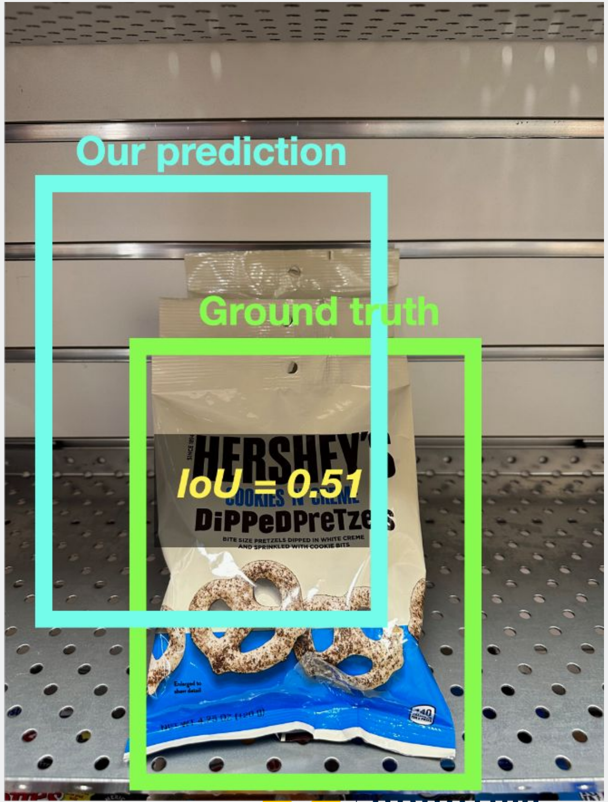
Comparing Boxes: IoU (Intersection over Union)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU) (Also called "Jaccard similarity" or "Jaccard index"):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

IoU > 0.5 is "decent",



MROBOTICS

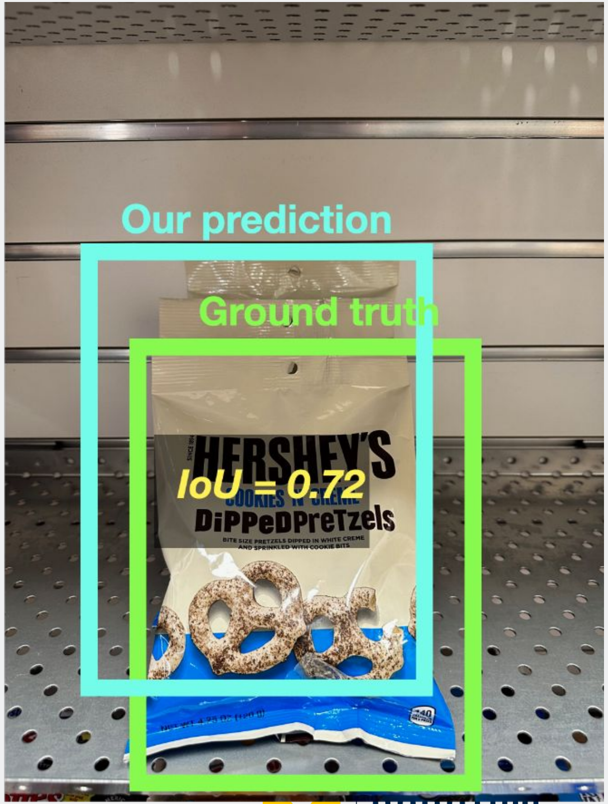
Comparing Boxes: IoU (Intersection over Union)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU) (Also called “Jaccard similarity” or “Jaccard index”):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

IoU > 0.5 is “decent”,
IoU > 0.7 is “pretty good”,



M | ROBOTICS

Comparing Boxes: IoU (Intersection over Union)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU) (Also called "Jaccard similarity" or "Jaccard index"):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

IoU > 0.5 is "decent",
IoU > 0.7 is "pretty good",
IoU > 0.9 is "almost perfect"

Detecting a single object

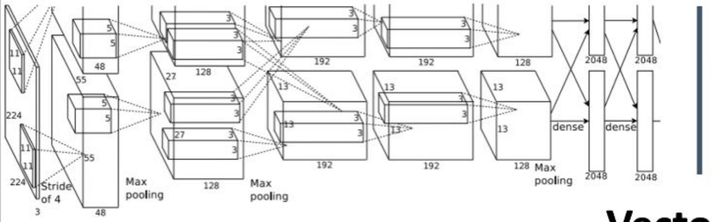


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector:
4096

Treat localization as a regression problem!

What??

Fully connected:
4096 to 10

Class scores:
Chocolate Pretzels: 0.9
Granola Bar: 0.02
Potato Chips: 0.02
Water Bottle: 0.02
Popcorn: 0.01
....

Correct Label:
Chocolate Pretzels

Softmax Loss

Multitask Loss

Weighted Sum → Loss

$$L = L_{cls} + \lambda L_{reg}$$

Fully connected:
4096 to 4

Box coordinates:
(x, y, w, h)

L2 Loss

Where??

Correct coordinates:
(x', y', w', h')

ROBOTICS

Detecting a single object

Often pretrained on ImageNet: Transfer learning

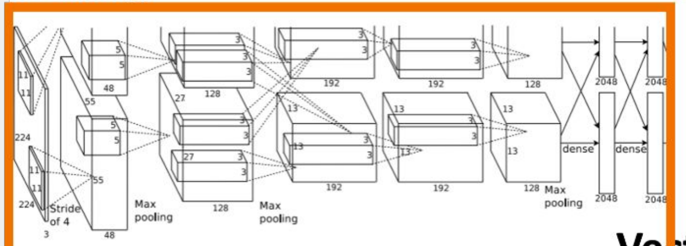


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector: 4096

Fully connected: 4096 to 10

What??

Class scores:
Chocolate Pretzels: 0.9
Granola Bar: 0.02
Potato Chips: 0.02
Water Bottle: 0.02
Popcorn: 0.01
....

Correct Label:
Chocolate Pretzels

Softmax Loss

Multitask Loss

Weighted Sum → Loss

$$L = L_{cls} + \lambda L_{reg}$$

Fully connected: 4096 to 4

Box coordinates:
(x, y, w, h)

L2 Loss

Correct coordinates:
(x', y', w', h')

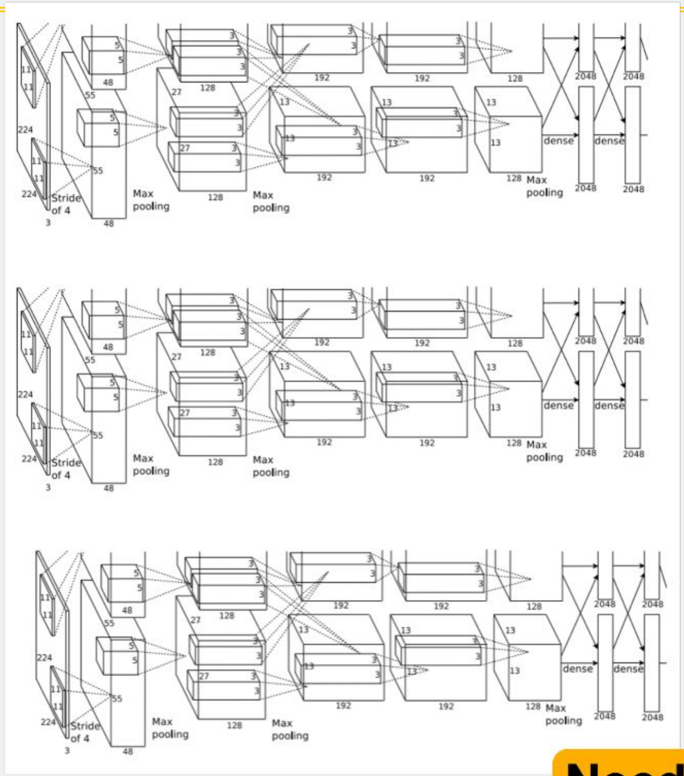
Where??

Treat localization as a regression problem!

Problem: Images can have more than one object!

ROBOTICS

Detecting Multiple Objects



Hershey's: (x, y, w, h)

4 numbers

Hershey's: (x, y, w, h)
Flipz: (x, y, w, h)
Reese's (x, y, w, h)

12 numbers

Chips: (x, y, w, h)
Chips: (x, y, w, h)
.....

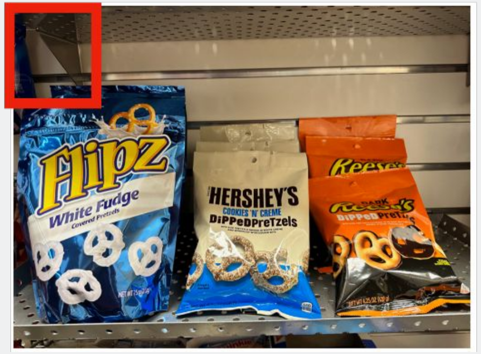
Many numbers!

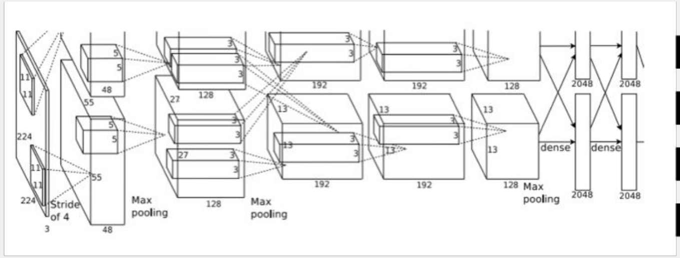
Need different numbers of output per image

ROBOTICS

Detecting Multiple Objects: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

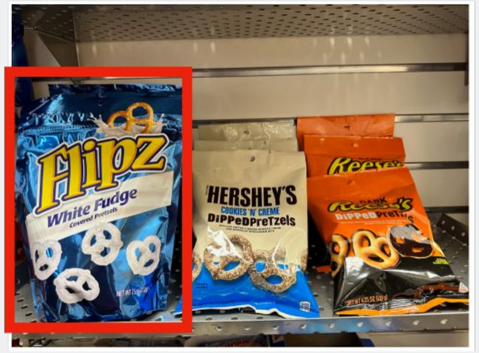


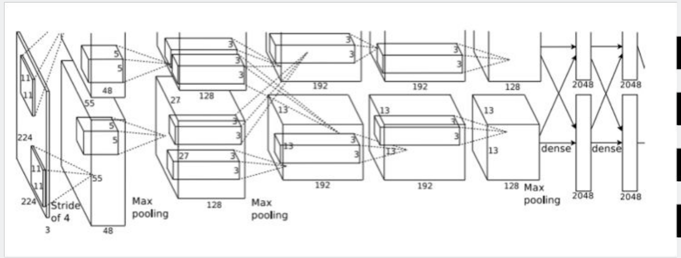


Hershey's: No
Flipz: No
Reese's: No
Background: Yes

Detecting Multiple Objects: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

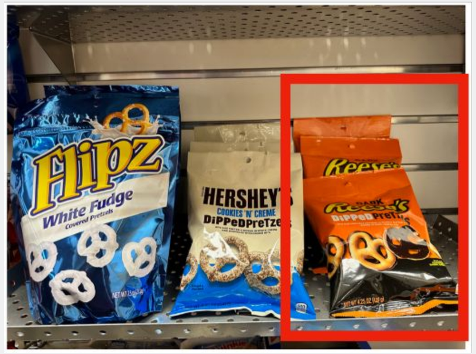


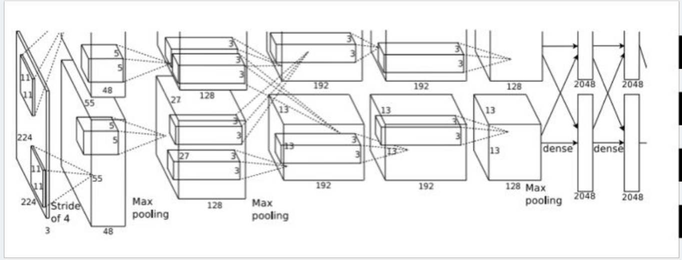


Hershey's: No
Flipz: Yes
Reese's: No
Background: No

Detecting Multiple Objects: Sliding Window

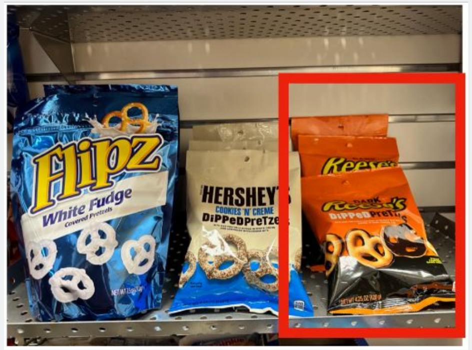
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background





Hershey's: No
Flipz: No
Reese's: Yes
Background: No

Detecting Multiple Objects: Sliding Window



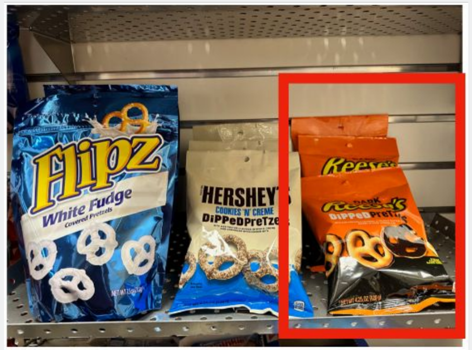
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

Q: How many possible boxes are there in a HxW image?

Aha Slides
(In-class participation)

https://ahaslides.com/EQCR8

Detecting Multiple Objects: Sliding Window

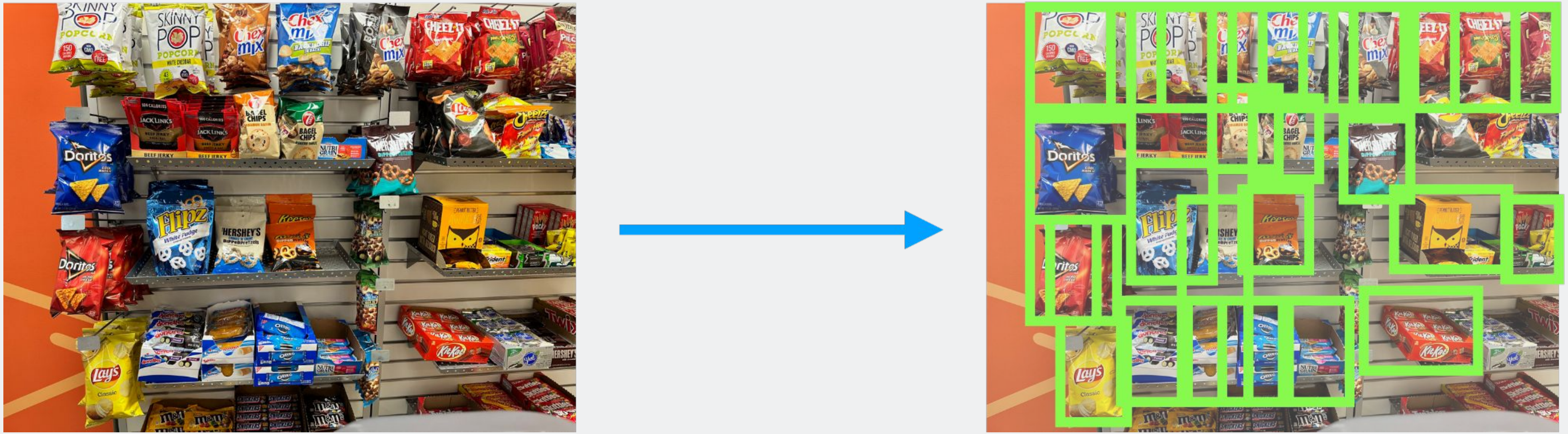


Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

800 x 600 image has ~58M boxes. No way we can evaluate them all

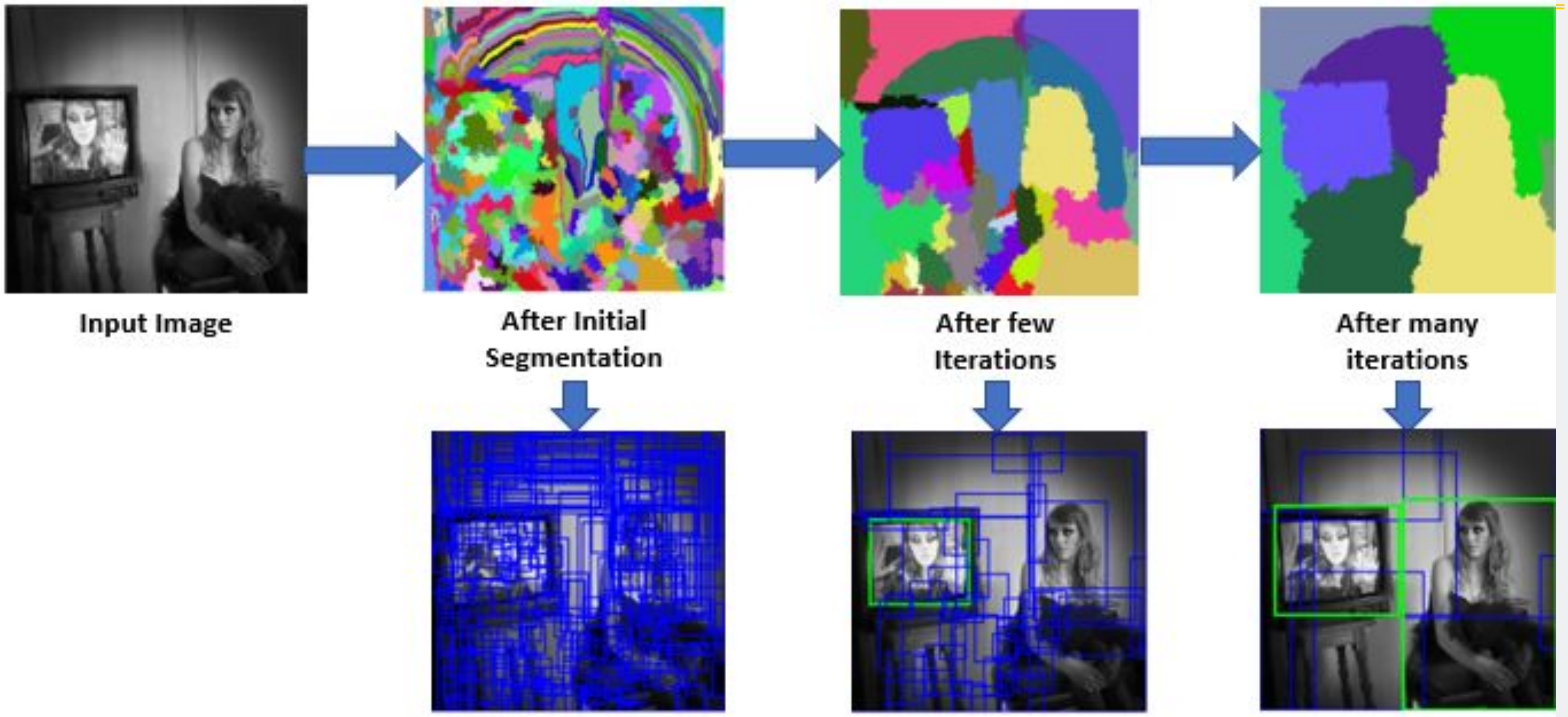
Region Proposals

- Find a small set of boxes that are likely to cover all objects
- Often based on heuristics: e.g. look for “blob-like” image regions
- Relatively fast to run; e.g. $\boxed{\text{Selective Search}}$ gives 2000 region proposals in a few seconds on CPU



Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014

*note on selective search



Felzenszwalb et al, “Efficient Graph-Based Image Segmentation“
https://www.geeksforgeeks.org/selective-search-for-object-detection-r-cnn/

ROBOTICS

Why do we still care about Region Proposals?

(CVPR 2024)

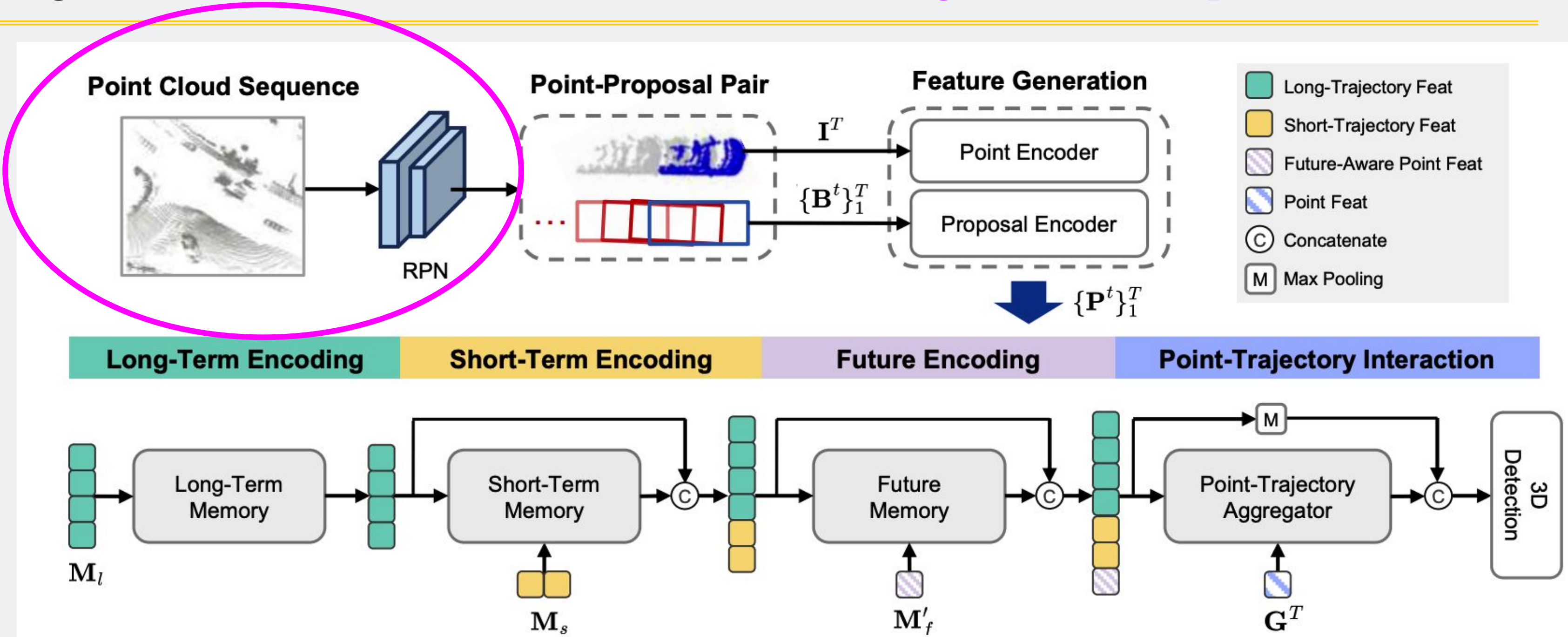
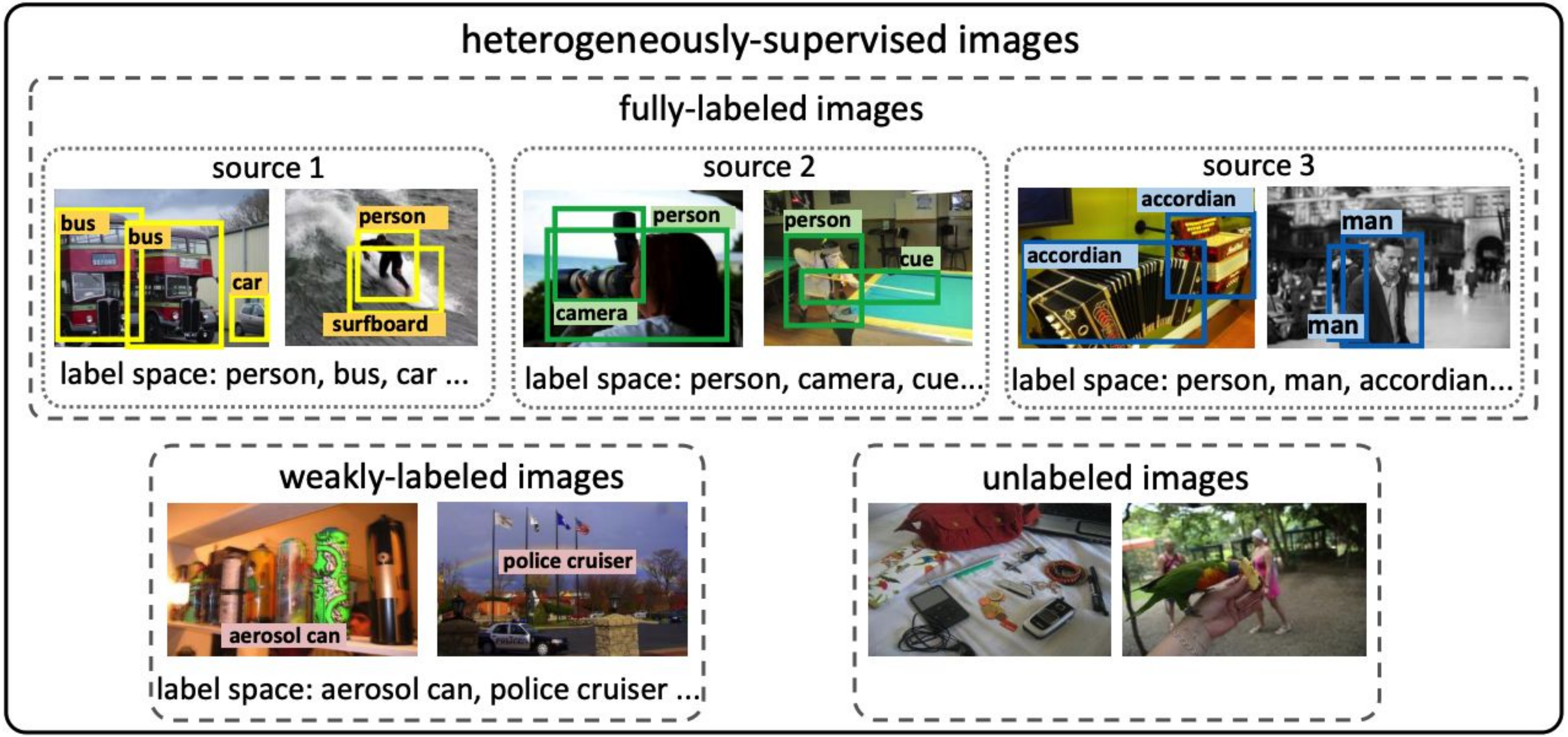


Figure 2. **Overall framework of the proposed Point-Trajectory Transformer (PTT).** First, we utilize a region proposal network (RPN) at timestamp T to generate proposals $\mathbf{B}^T$ for each frame, sample the corresponding point-of-interest $\mathbf{I}^T$, and connect past T-frame 3D proposals to form proposal trajectories $\{\mathbf{B}^1, ..., \mathbf{B}^T\}$. Then, we take the single-frame point cloud for each object and its previous multi-

https://openaccess.thecvf.com/content/CVPR2024/papers/Huang_PTT_Point-Trajectory_Transformer_for_Efficient_Temporal_3D_Object_Detection_CVPR_2024_paper.pdf

Why do we still care about Region Proposals?

(TPAMI 2024)



https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10552883

"UniDetector"

ROBOTICS

Why do we still care about Region Proposals?

(TPAMI 2024, cont'd)

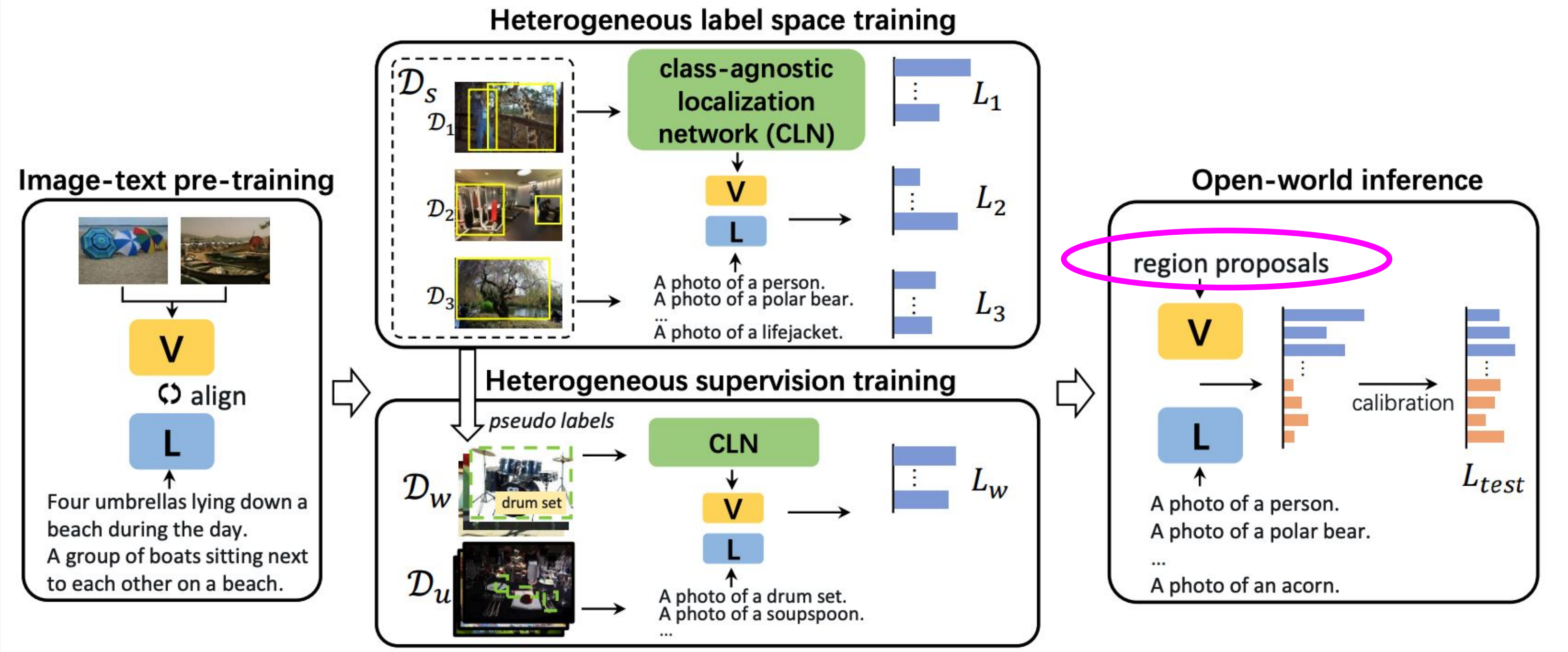


Fig. 2: **Overview of UniDetector.** It consists of four steps. With the image-text pre-training parameters, UniDetector is

https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10552883

"UniDetector"

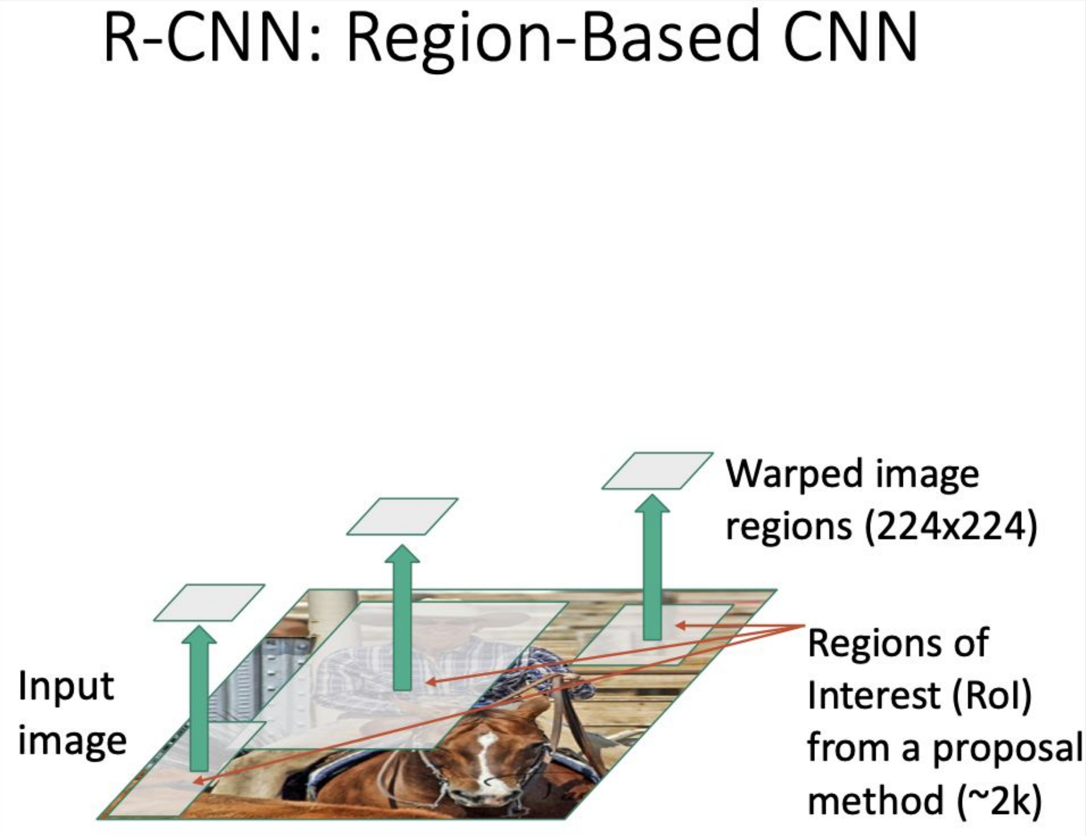
R-CNN: Region-Based CNN

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission

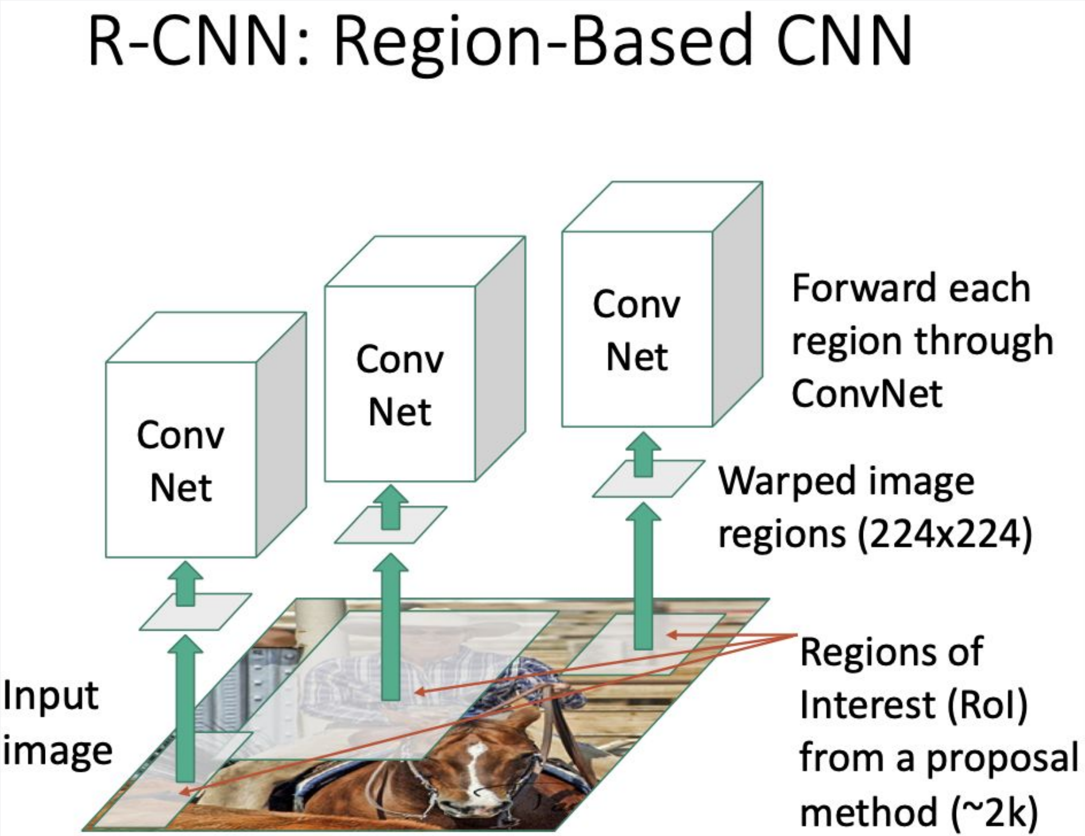
R-CNN: Region-Based CNN

R-CNN: Region-Based CNN

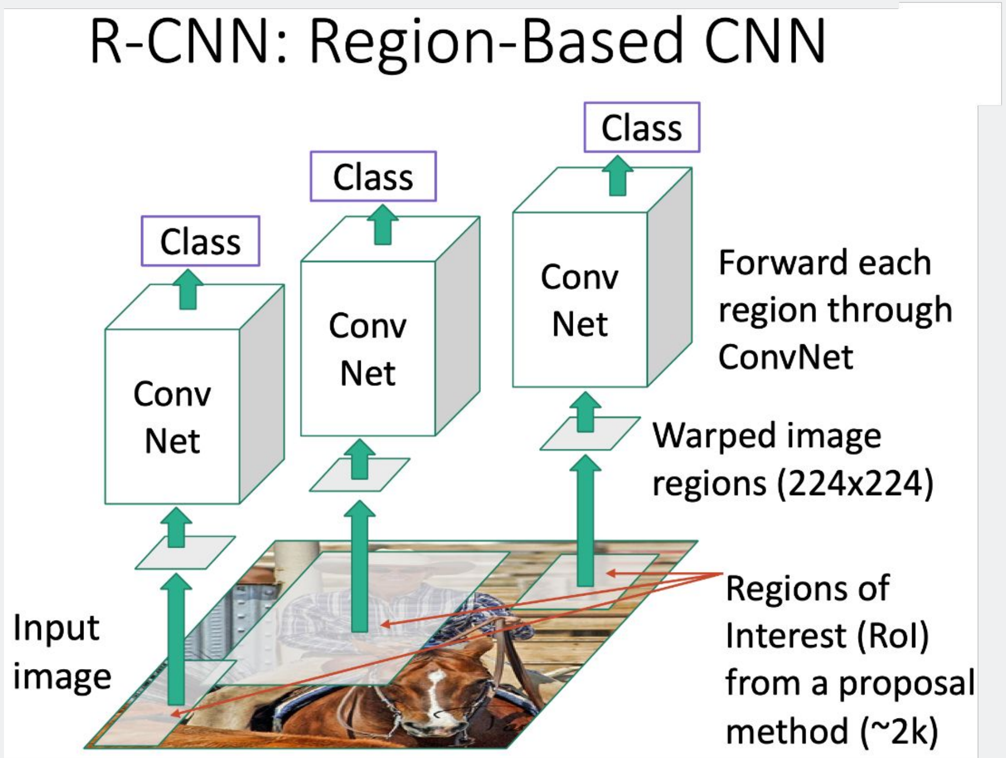
R-CNN: Region-Based CNN



R-CNN: Region-Based CNN

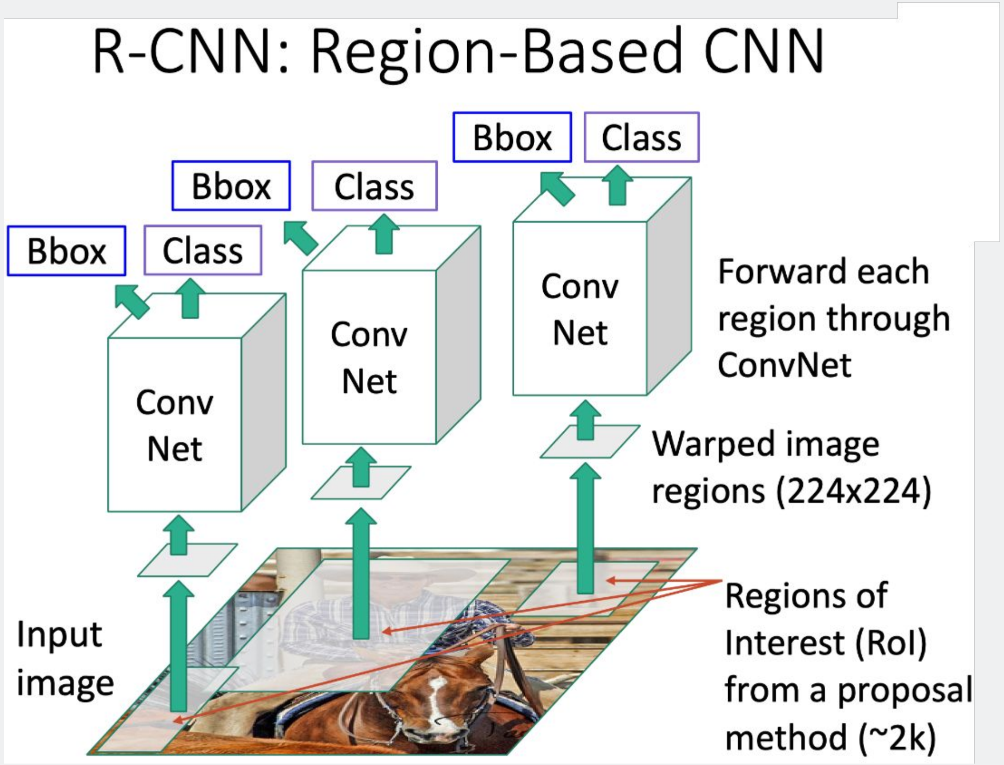


R-CNN: Region-Based CNN



Classify each region

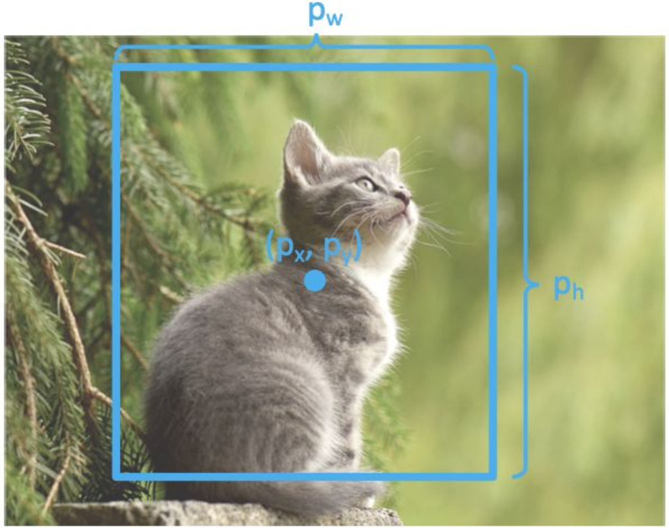
R-CNN: Region-Based CNN



Classify each region

Bounding box regression:
Predict "transform" to correct the RoI: 4 numbers (t_x, t_y, t_h, t_w)

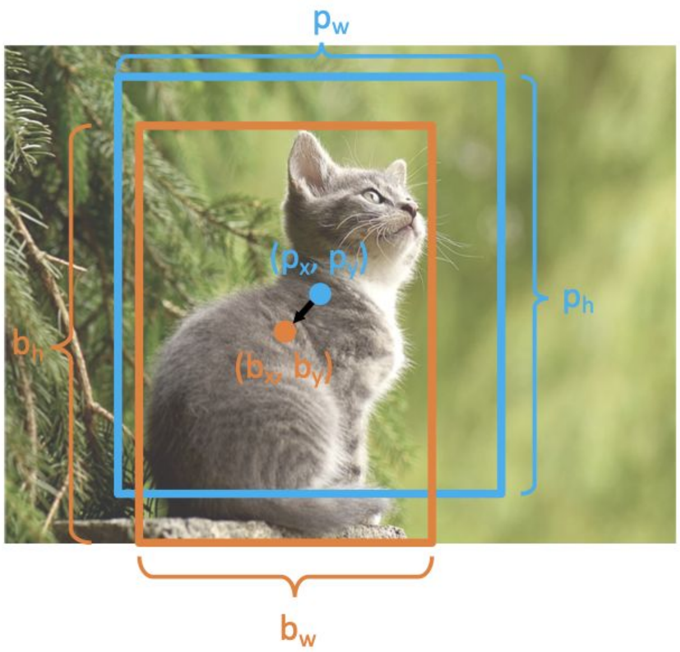
R-CNN: Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

R-CNN: Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

The output box is defined by:

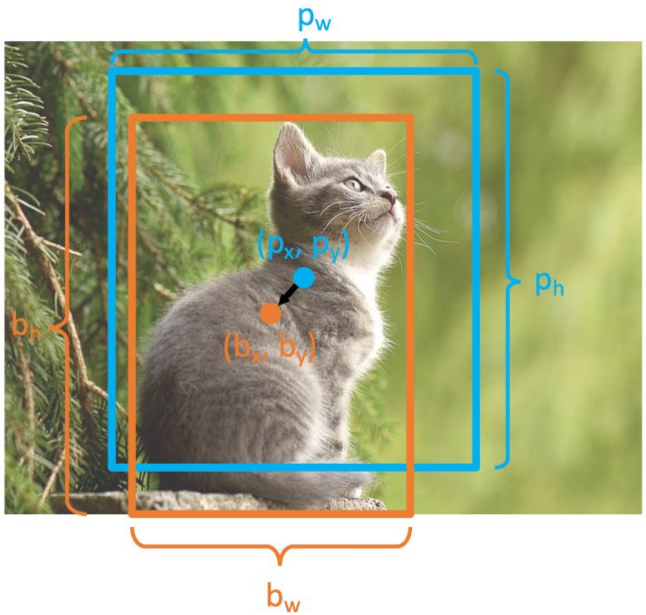
$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$

Shift center by amount relative to proposal size

$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

Scale proposal; exp ensures that scaling factor is > 0

R-CNN: Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

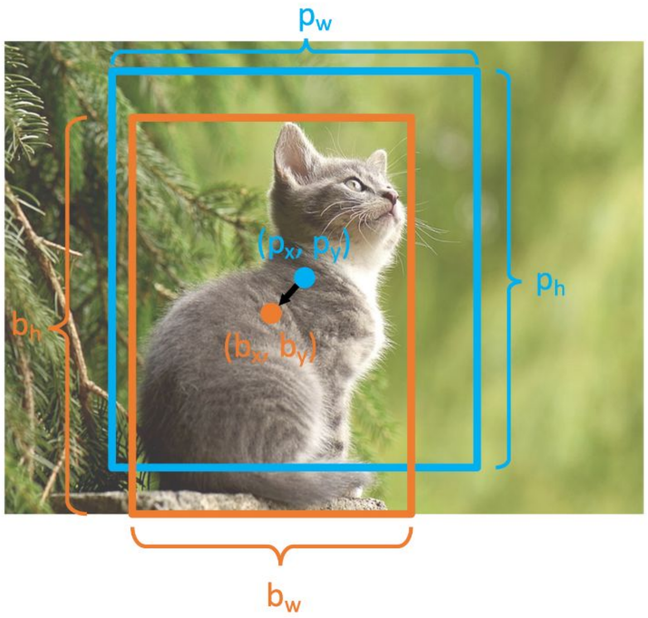
The output box is defined by:

$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$
$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

When transform is 0, output = proposal

L2 regularization encourages leaving proposal unchanged

R-CNN: Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

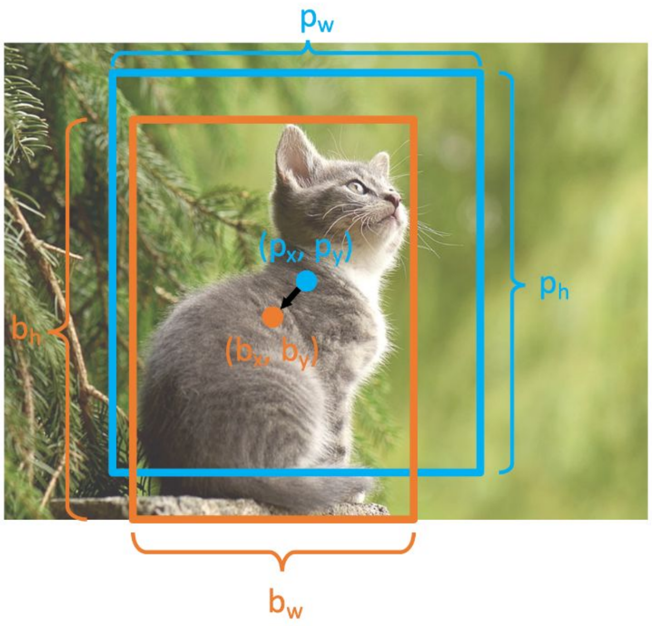
Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

The output box is defined by:

$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$
$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

Scale / Translation invariance: Transform encodes *relative* difference between proposal and output; important since CNN doesn't see absolute size or position after cropping

R-CNN: Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

The output box is defined by

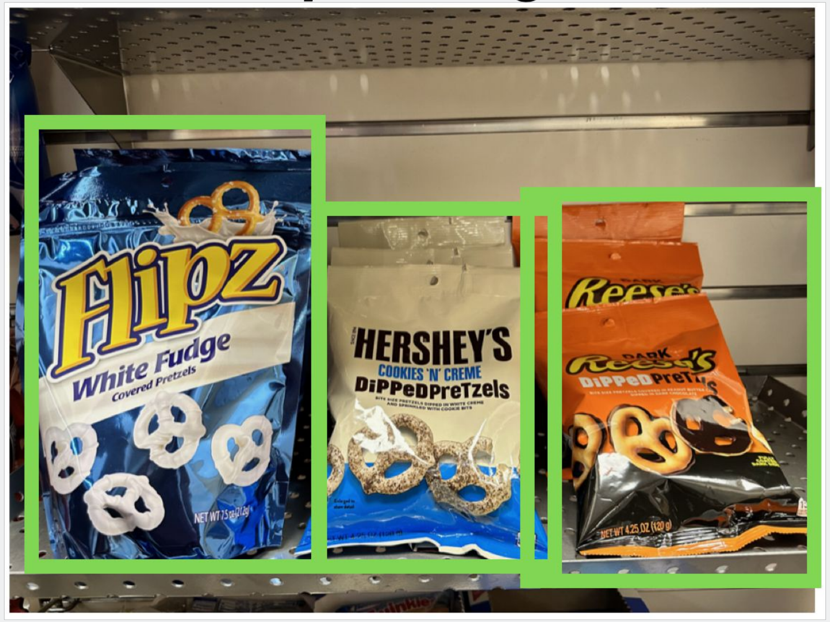
$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$
$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

Given proposal and target output, we can solve for the transform the network should output:

$$t_x = (b_x - p_x)/p_w$$
$$t_y = (b_y - p_y)/p_h$$
$$t_w = \log(b_w/p_w)$$
$$t_h = \log(b_h/p_h)$$

R-CNN: Training

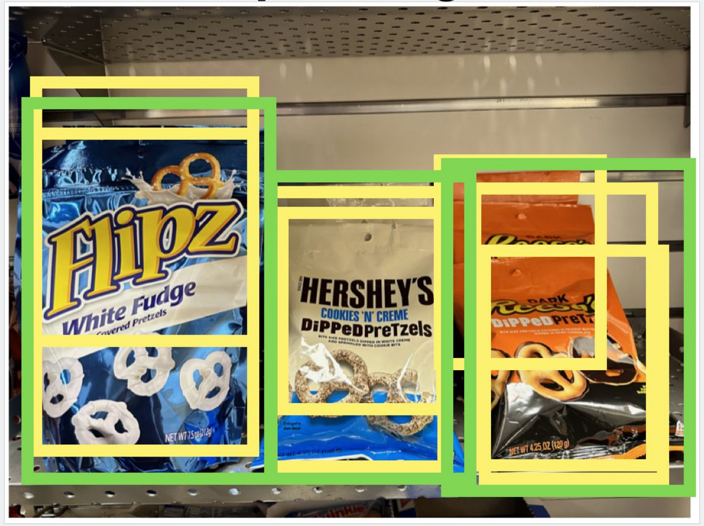
Input Image



Ground Truth

R-CNN: Training

Input Image

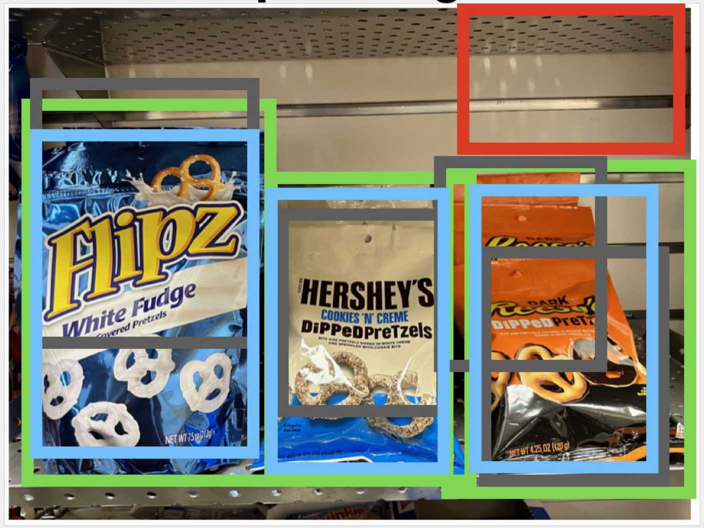


Ground Truth

Region Proposals

R-CNN: Training

Input Image

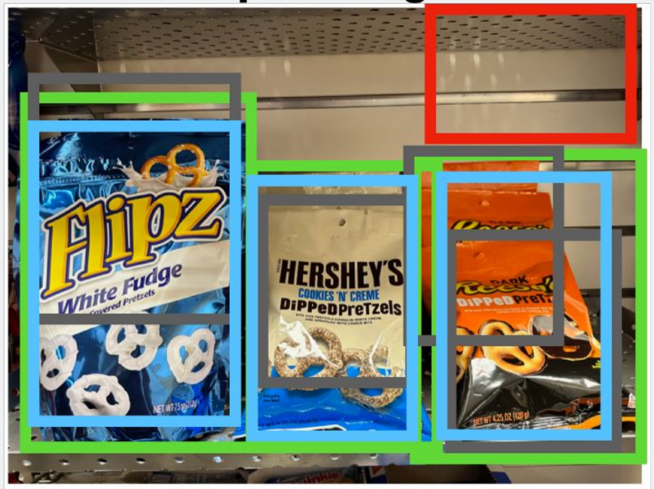


Ground Truth | Positive

Neutral | Negative

R-CNN: Training

Input Image



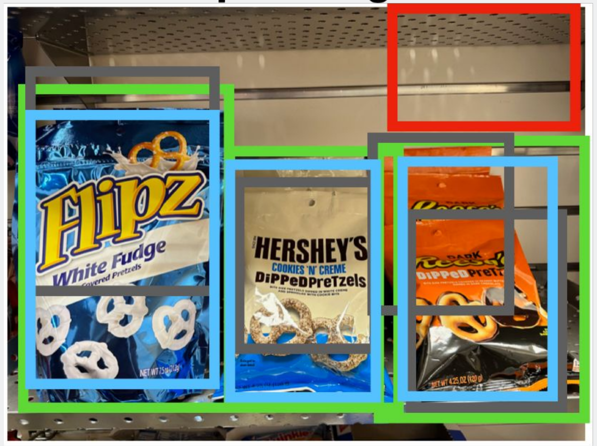
Ground Truth	Positive
Neutral	Negative

Categorize each region proposal as positive, negative or neutral based on overlap with the Ground truth boxes:

Positive: > 0.5 IoU with a GT box
Negative: < 0.3 IoU with all GT boxes
Neutral: between 0.3 and 0.5 IoU with GT boxes

R-CNN: Training

Input Image



Ground Truth	Positive
Neutral	Negative

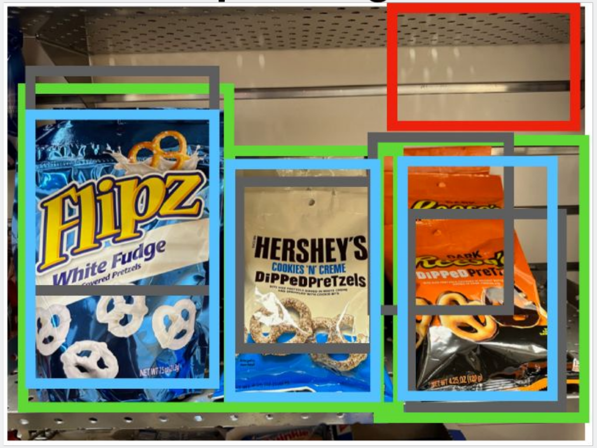
Crop pixels from each positive and negative proposal, resize to 224 x 224

Run each region through CNN
Positive regions: predict class and transform
Negative regions: just predict class

ROBOTICS

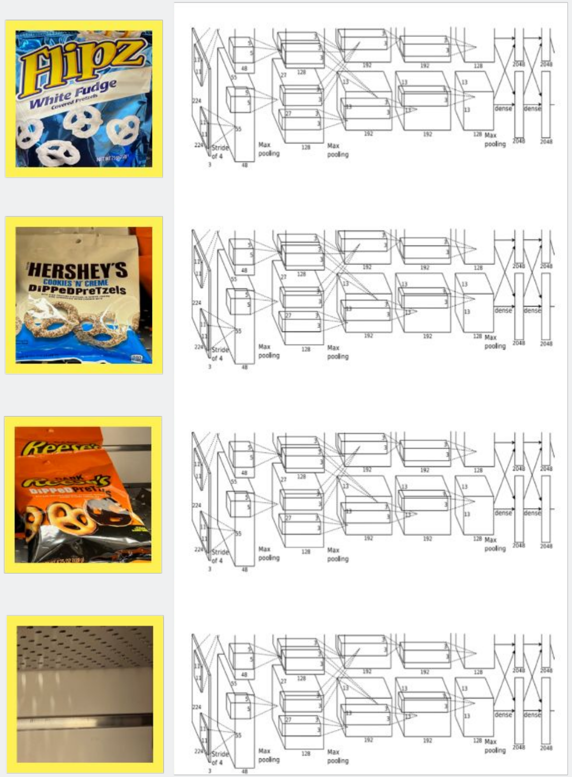
R-CNN: Training

Input Image



Ground Truth	Positive
Neutral	Negative

Run each region through CNN
Positive regions: predict class and transform
Negative regions: just predict class

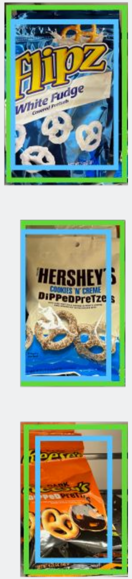


Class target: Flipz
Box target: →

Class target: Hershey's
Box target: →

Class target: Reese's
Box target: →

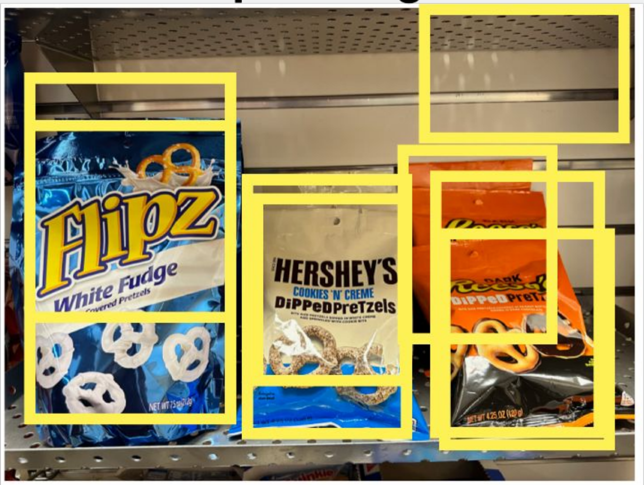
Class target: Background
Box target: None



M | ROBOTICS

R-CNN: Test Time

Input Image



Region Proposals

<u>Run proposal method:</u>

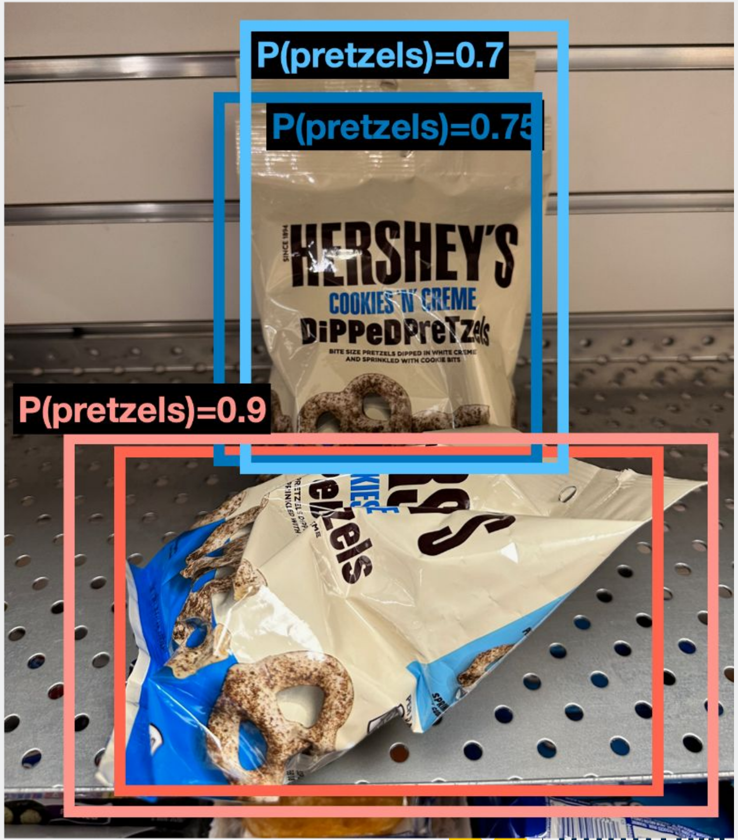
1. Run CNN on each proposal to get class scores, transforms
2. Threshold class scores to get a set of detections

2 Problems:

1. CNN often outputs overlapping boxes
2. How to set thresholds?

Overlapping Boxes

Problem: Object detectors often output many overlapping detections

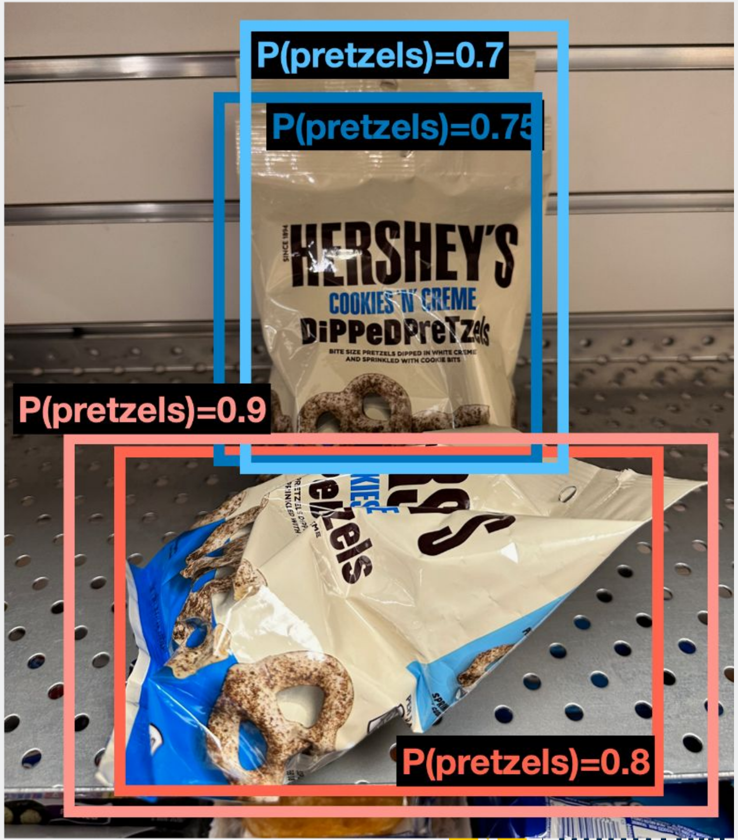


Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections

Solution: Post-process raw detections using Non-Max Suppression (NMS)

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU> threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1



Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections

Solution: Post-process raw detections using Non-Max Suppression (NMS)

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU> threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

IoU(,) = 0.8
IoU(,) = 0.03
IoU(,) = 0.05

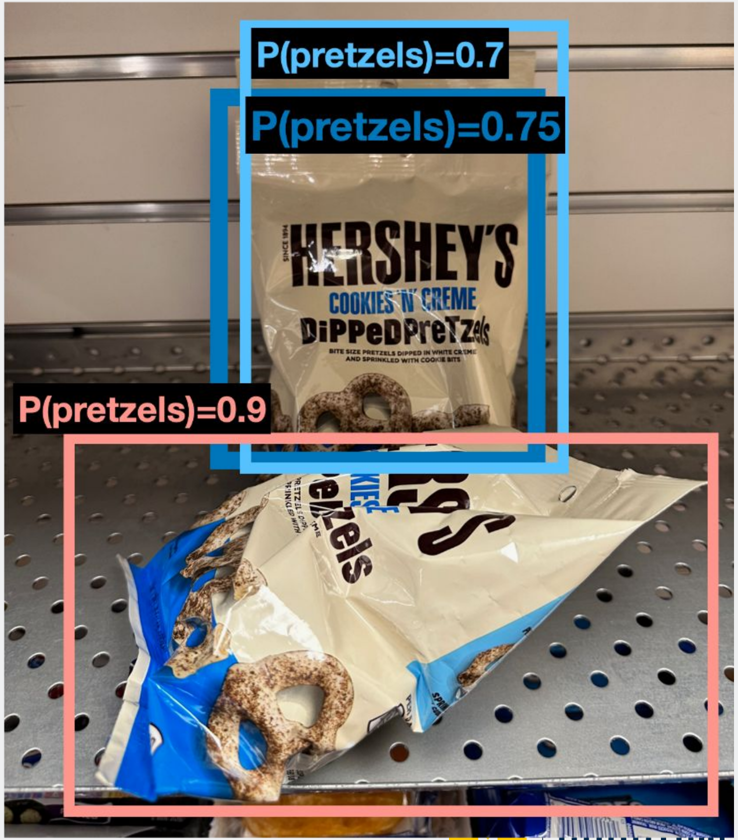
Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections

Solution: Post-process raw detections using Non-Max Suppression (NMS)

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU> threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

IoU(■,■) = 0.85

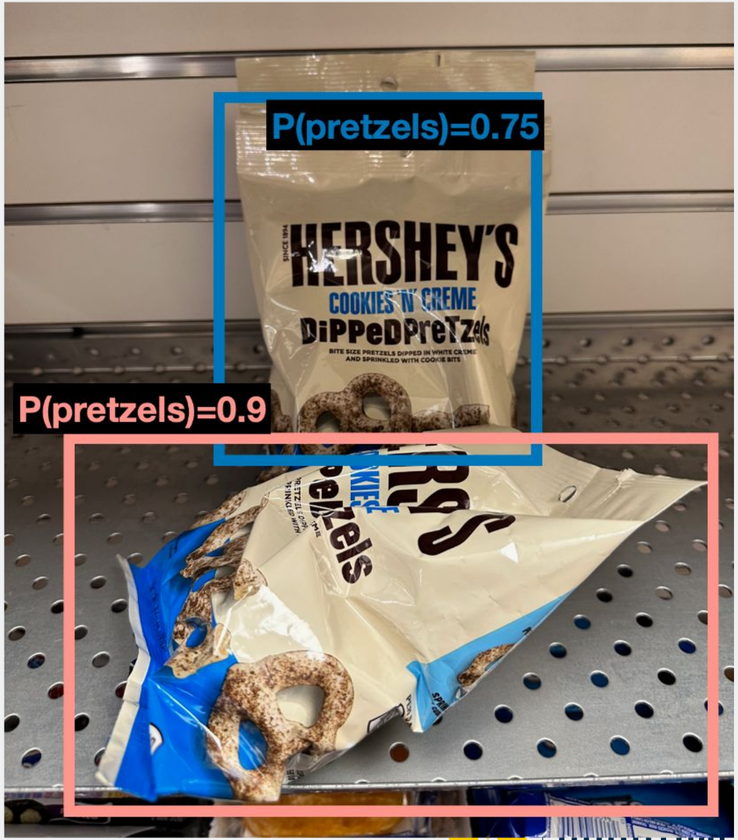


Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections

Solution: Post-process raw detections using Non-Max Suppression (NMS)

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU> threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1



Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections

Solution: Post-process raw detections using Non-Max Suppression (NMS)

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU> threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

Problem: NMS may eliminate "good" boxes when objects are highly overlapping… no good solution

Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)

Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)

All pretzel detections sorted by score

All ground-truth pretzel boxes

Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative

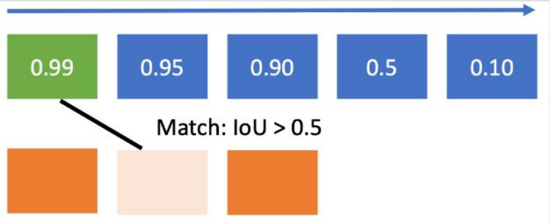
All pretzel detections sorted by score

All ground-truth pretzel boxes

Evaluating Object Detectors: Mean Average Precision (mAP)

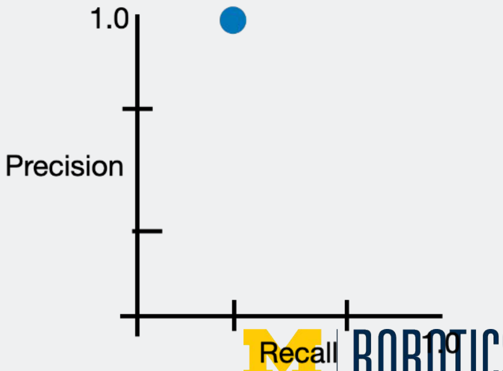
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
   1. For each detection (highest score to lowest score)
      1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      2. Otherwise mark it as negative
      3. Plot a point on PR curve

All pretzel detections sorted by score



All ground-truth pretzel boxes

Precision = 1/1 = 1.0
Recall = 1/3 = 0.33



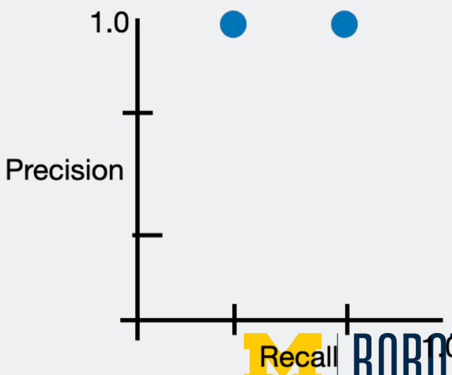
Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR curve

All pretzel detections sorted by score

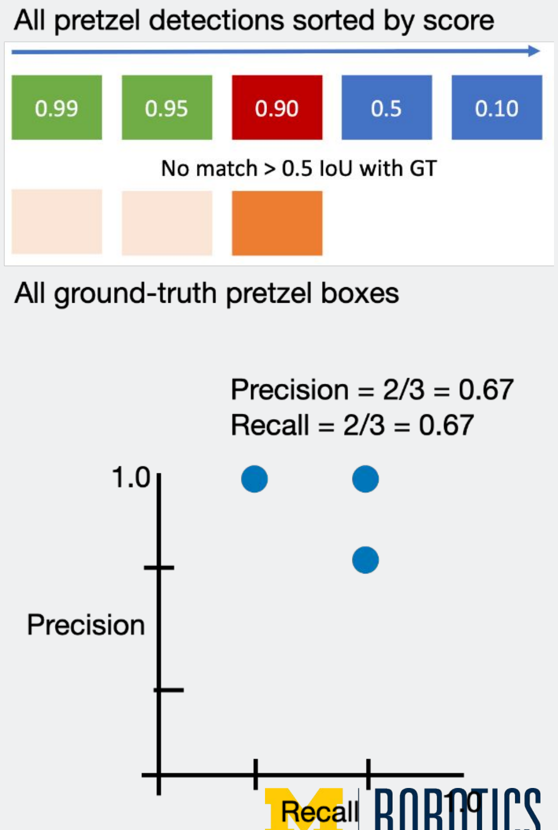
All ground-truth pretzel boxes

Precision = 2/2 = 1.0
Recall = 2/3 = 0.67



Evaluating Object Detectors: Mean Average Precision (mAP)

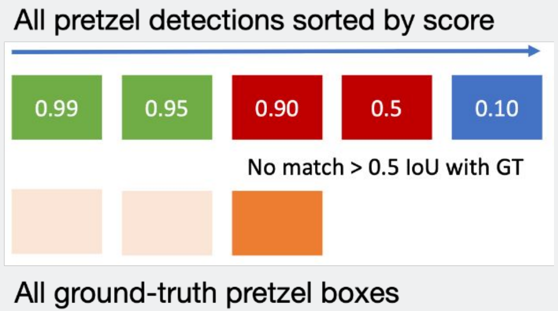
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
   1. For each detection (highest score to lowest score)
      1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      2. Otherwise mark it as negative
      3. Plot a point on PR curve

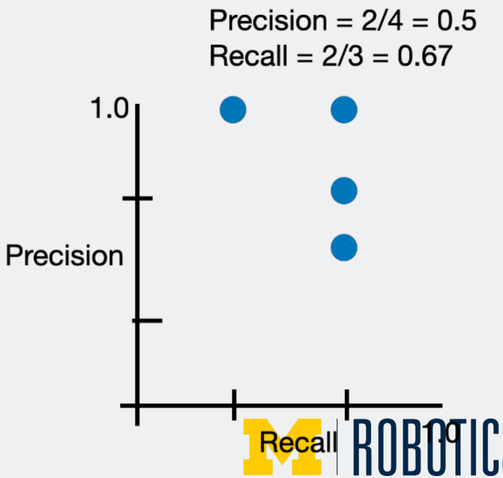


ROBOTICS

Evaluating Object Detectors: Mean Average Precision (mAP)

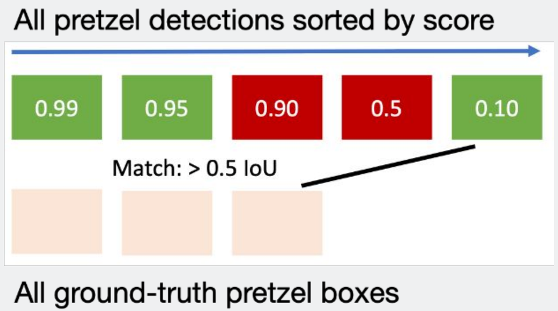
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR curve

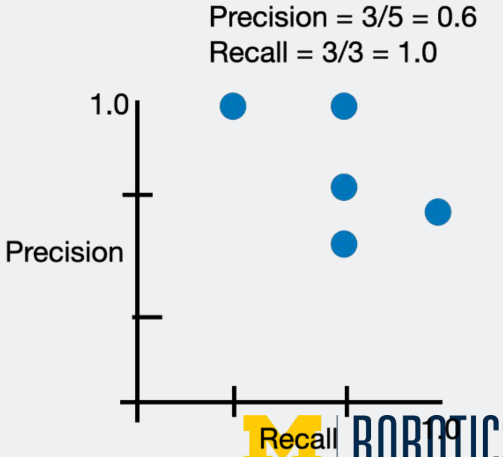




Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR curve



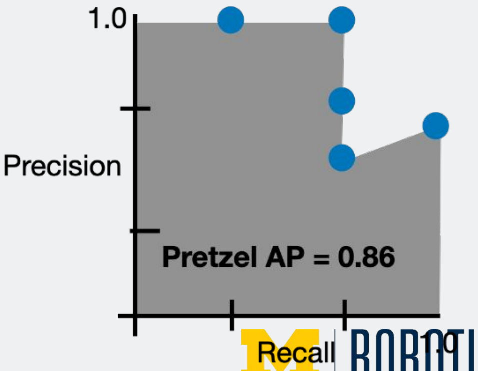


Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR curve
    2. Average Precision (AP) = area under PR curve

All pretzel detections sorted by score

All ground-truth pretzel boxes



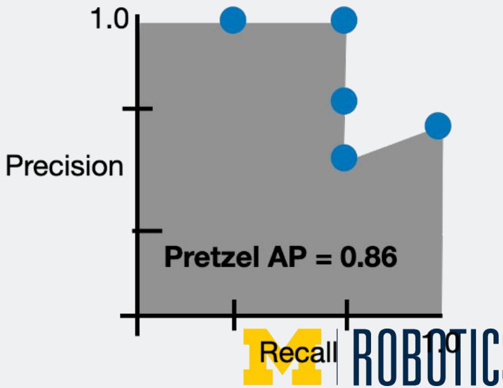
Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR curve
    2. Average Precision (AP) = area under PR curve

How to get AP = 1.0: Hit all GT boxes with IoU > 0.5, and have no "false positive" detections ranked above any "true positives"

All pretzel detections sorted by score

All ground-truth pretzel boxes



Evaluating Object Detectors: Mean Average Precision (mAP)

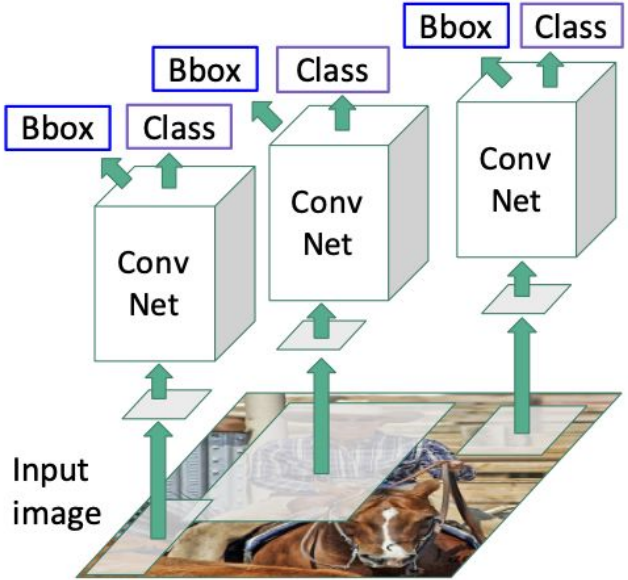
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
   1. For each detection (highest score to lowest score)
      1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      2. Otherwise mark it as negative
      3. Plot a point on PR curve
   2. Average Precision (AP) = area under PR curve
3. Mean Average Precision (mAP) = average of AP for each category

Flipz AP = 0.60
Hershey's AP = 0.85
Reese's AP = 0.81
mAP@0.5 = 0.75

Fast R-CNN

“Slow” R-CNN

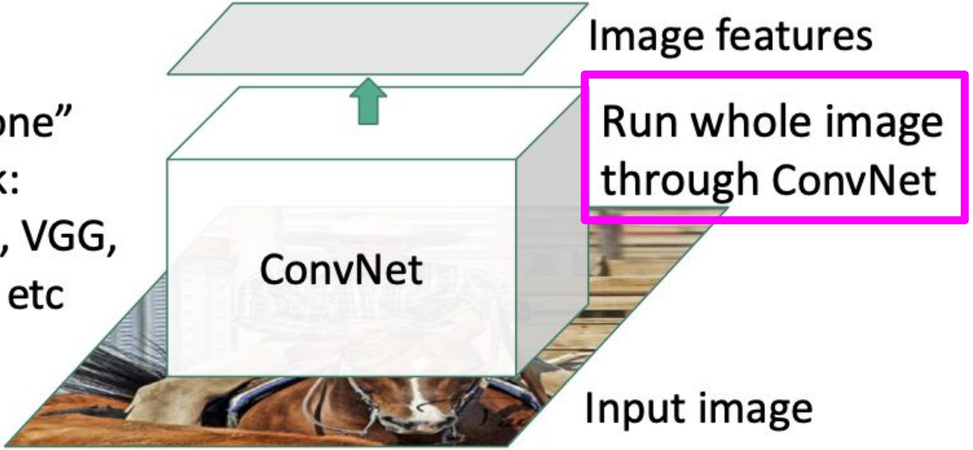
Process each region independently



Fast R-CNN

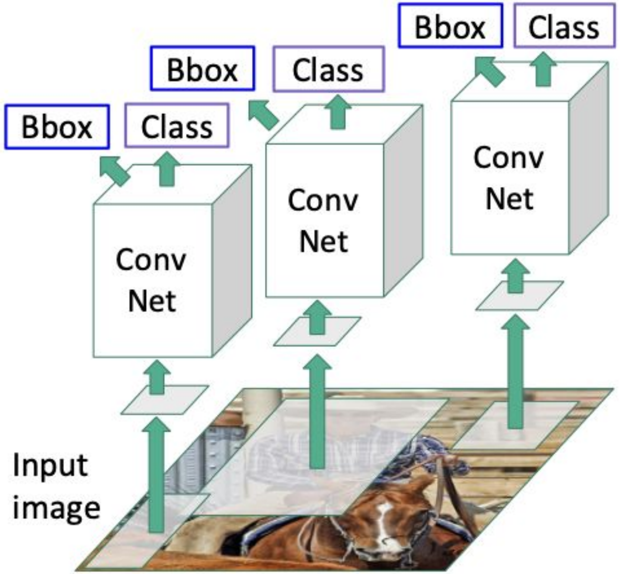
“Heavy duty backbone”

“Backbone” network: AlexNet, VGG, ResNet, etc

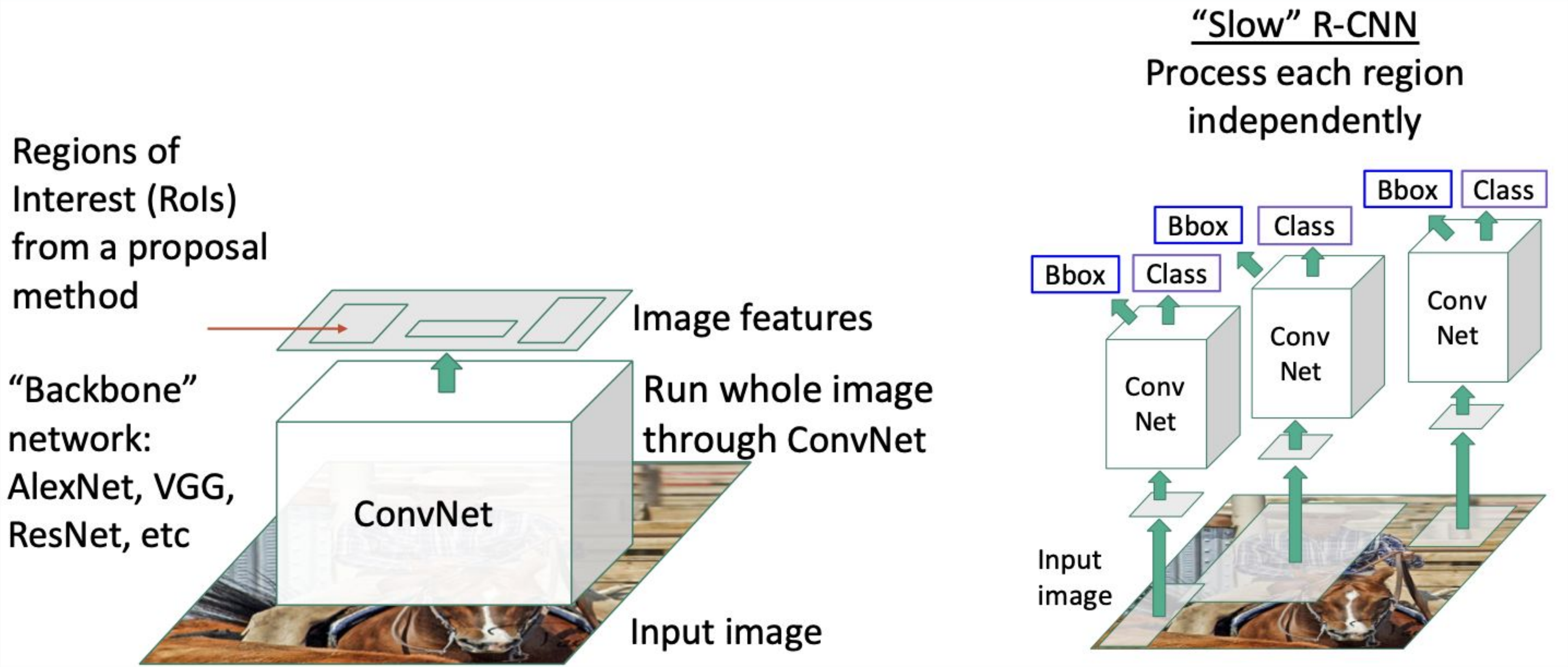


“Slow” R-CNN

Process each region independently

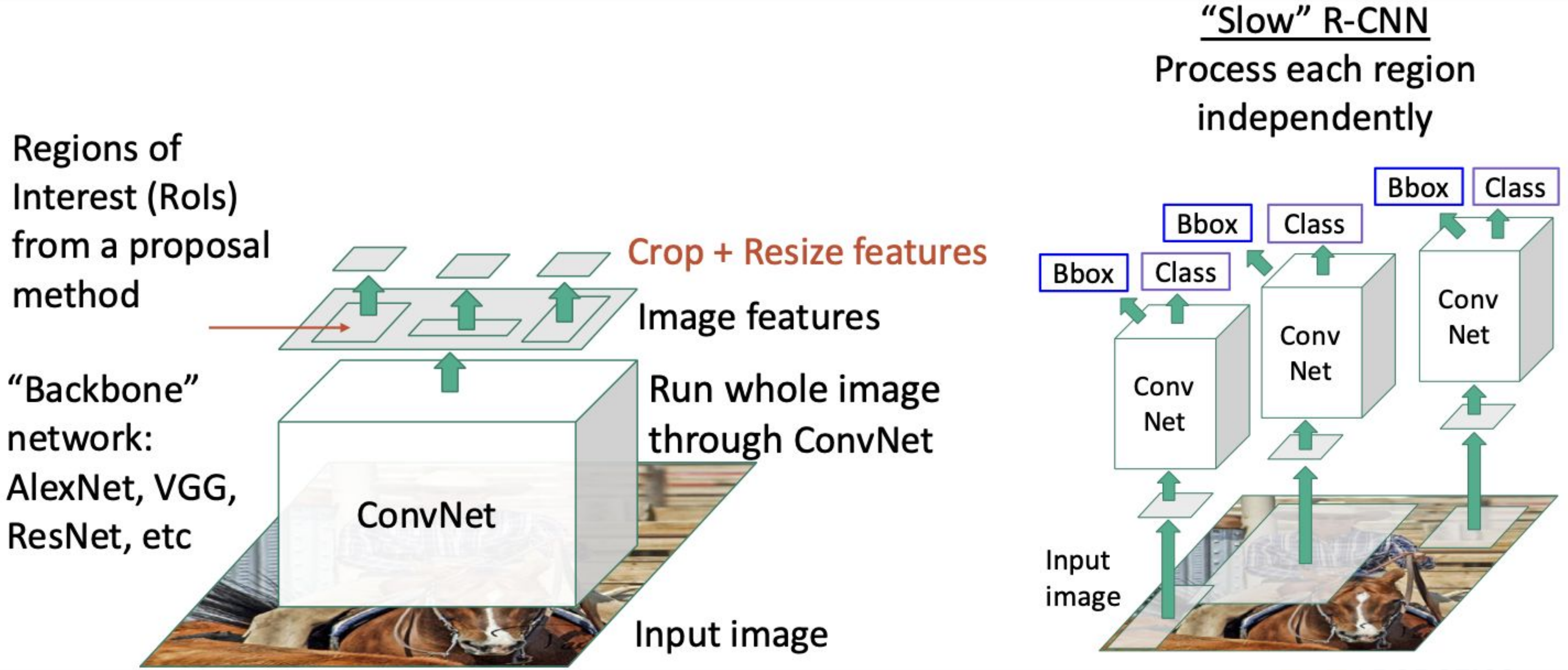


Fast R-CNN

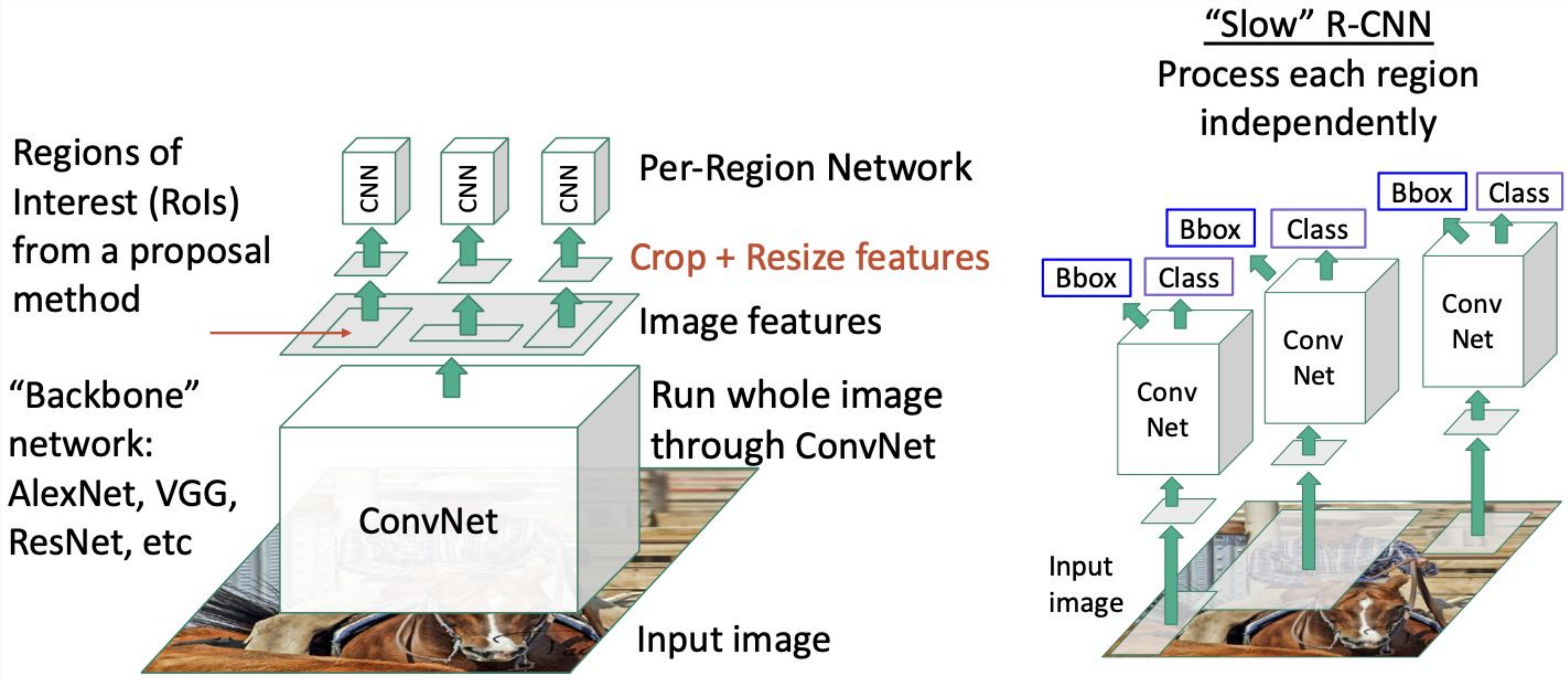


"Slow" R-CNN
Process each region independently
Regions of Interest (RoIs) from a proposal method
"Backbone" network: AlexNet, VGG, ResNet, etc
Image features
Run whole image through ConvNet
ConvNet
Input image
Bbox
Class
Bbox
Class
Bbox
Class
Conv Net
Conv Net
Conv Net
Input image

Fast R-CNN

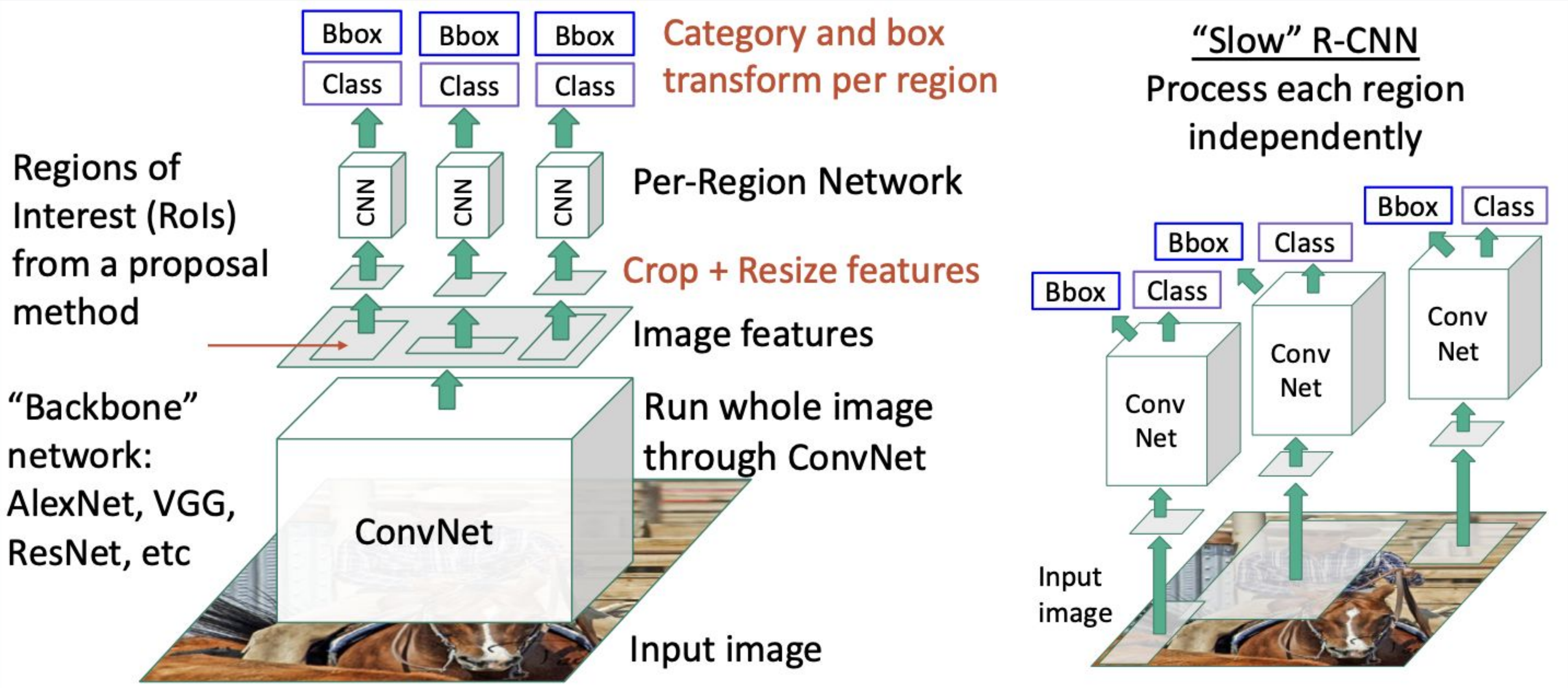

"Slow" R-CNN
Process each region independently
Regions of Interest (RoIs) from a proposal method
"Backbone" network: AlexNet, VGG, ResNet, etc
Crop + Resize features
Image features
Run whole image through ConvNet
ConvNet
Input image
Bbox
Class
Bbox
Class
Bbox
Class
Conv Net
Conv Net
Conv Net
Input image

Fast R-CNN

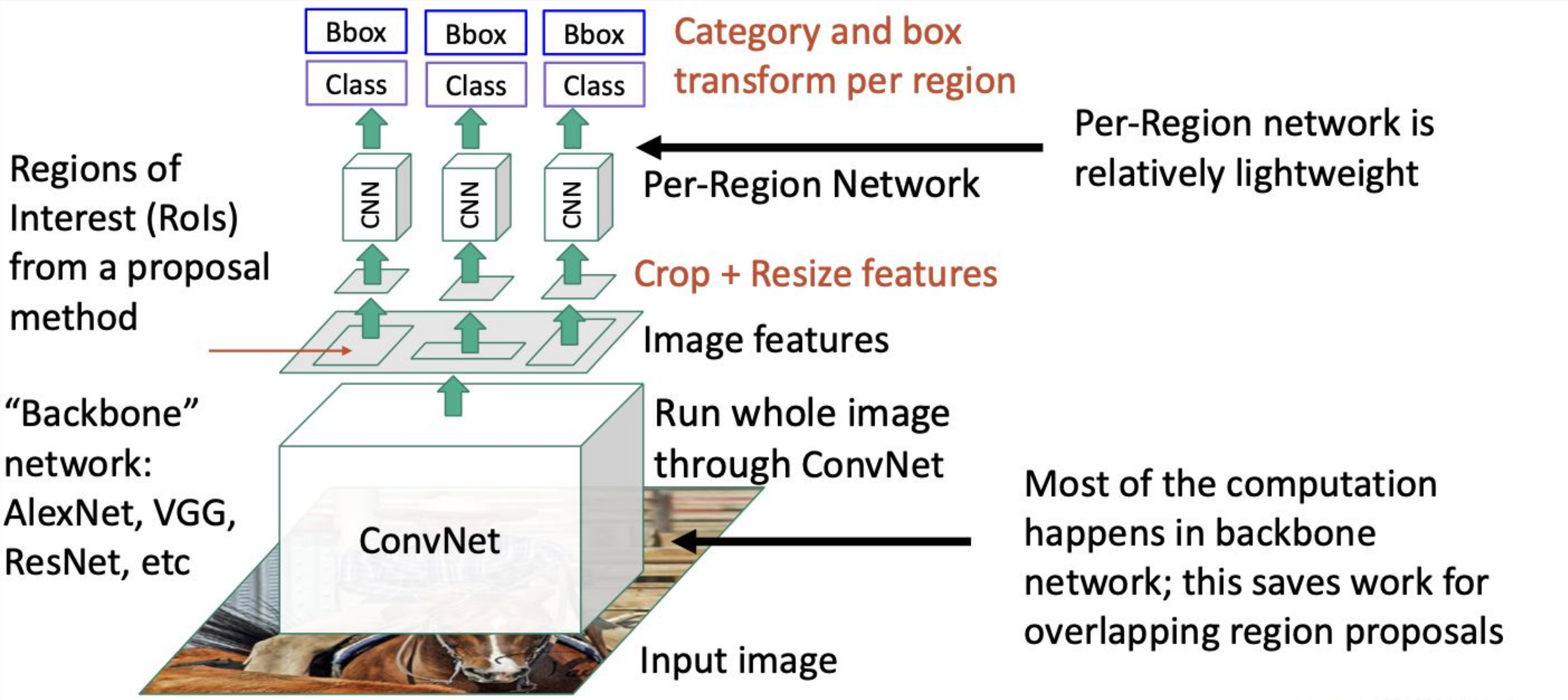


"Slow" R-CNN
Process each region independently
Regions of Interest (RoIs) from a proposal method
"Backbone" network: AlexNet, VGG, ResNet, etc
CNN
CNN
CNN
Per-Region Network
Crop + Resize features
Image features
Run whole image through ConvNet
ConvNet
Input image
Bbox
Class
Bbox
Class
Bbox
Class
Conv Net
Conv Net
Conv Net
Input image

Fast R-CNN

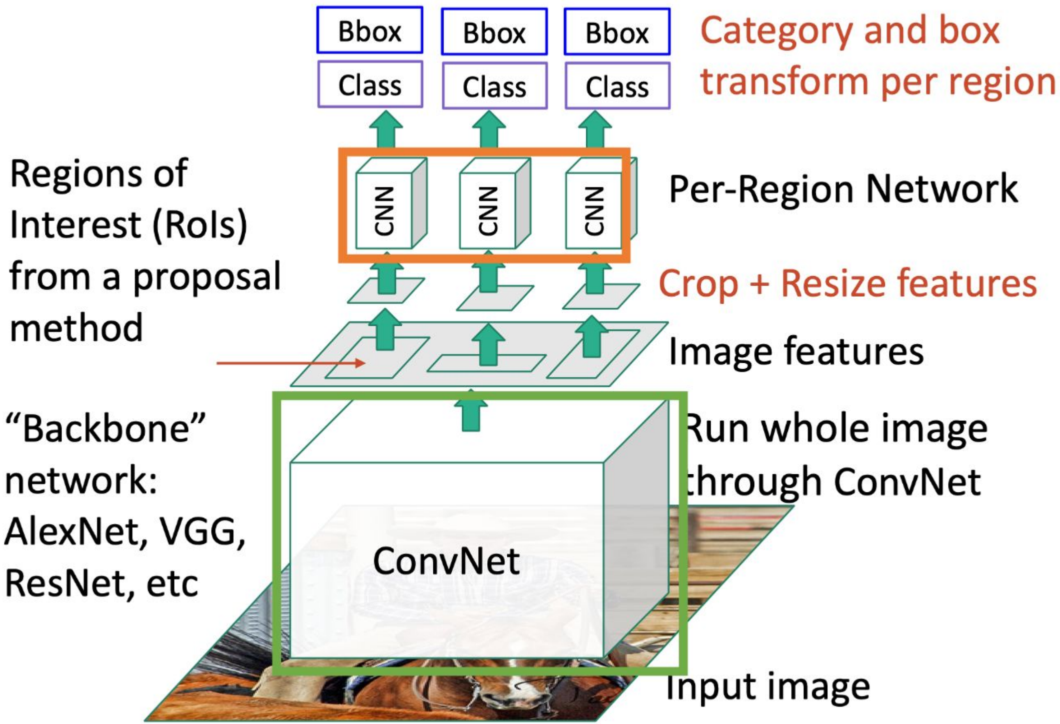

Bbox
Class
Bbox
Class
Bbox
Class
Category and box transform per region
CNN
CNN
CNN
Per-Region Network
Crop + Resize features
Image features
Regions of Interest (RoIs) from a proposal method
"Backbone" network: AlexNet, VGG, ResNet, etc
ConvNet
Run whole image through ConvNet
Input image
"Slow" R-CNN
Process each region independently
Bbox
Class
Bbox
Class
Bbox
Class
Conv Net
Conv Net
Conv Net
Input image

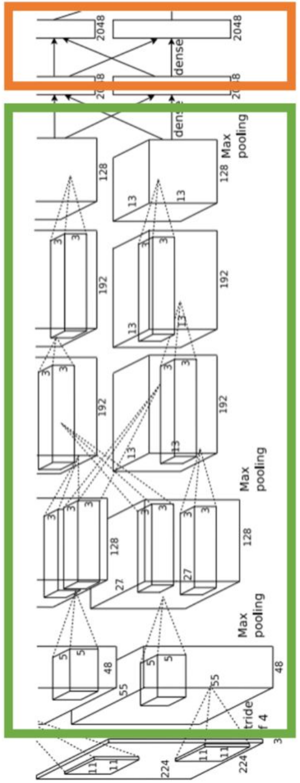
Fast R-CNN



Bbox
Bbox
Bbox
Class
Class
Class
Category and box transform per region
Per-Region network is relatively lightweight
Regions of Interest (RoIs) from a proposal method
CNN
CNN
CNN
Per-Region Network
Crop + Resize features
Image features
"Backbone" network: AlexNet, VGG, ResNet, etc
Run whole image through ConvNet
ConvNet
Most of the computation happens in backbone network; this saves work for overlapping region proposals
Input image

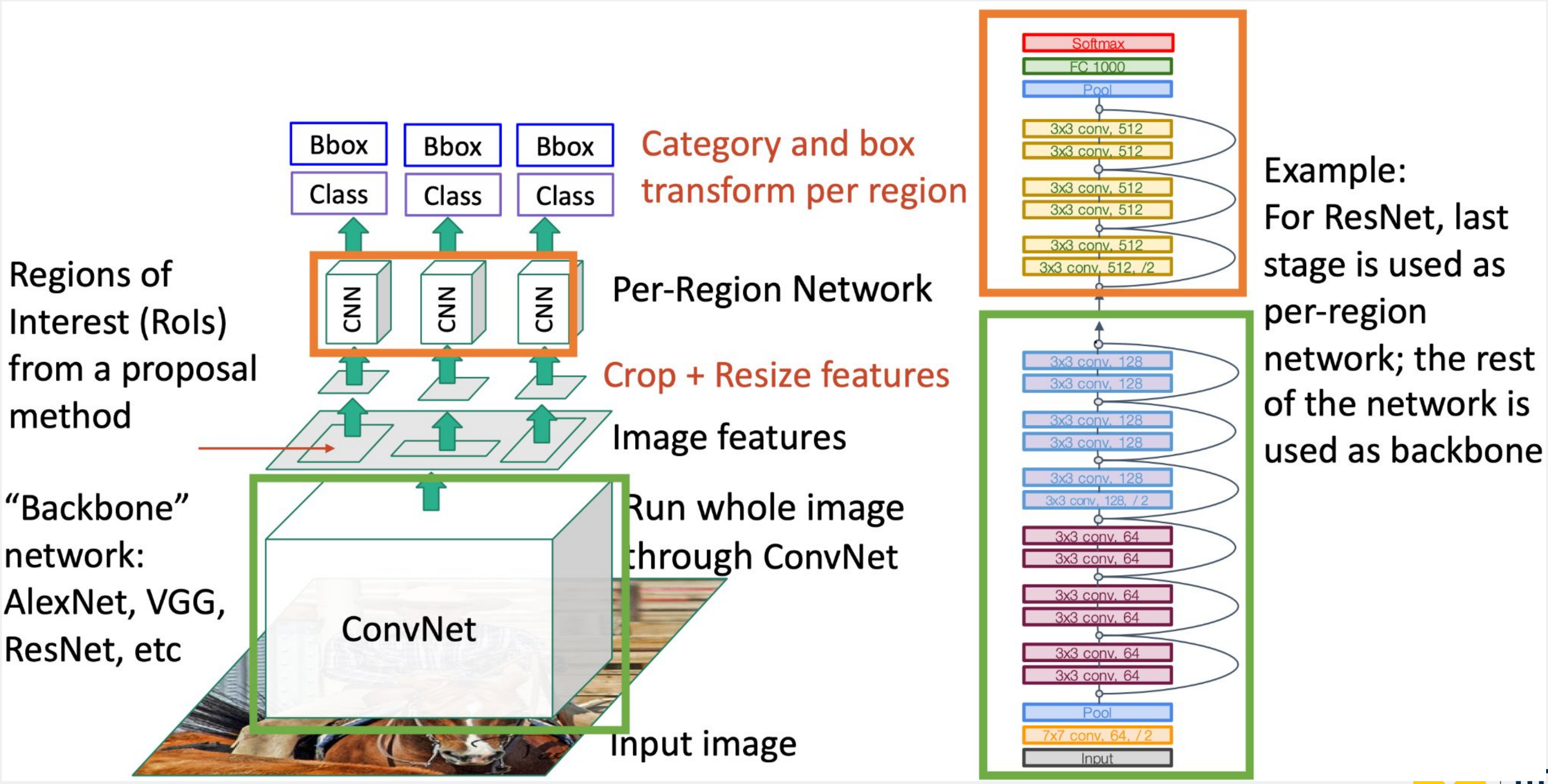
Fast R-CNN





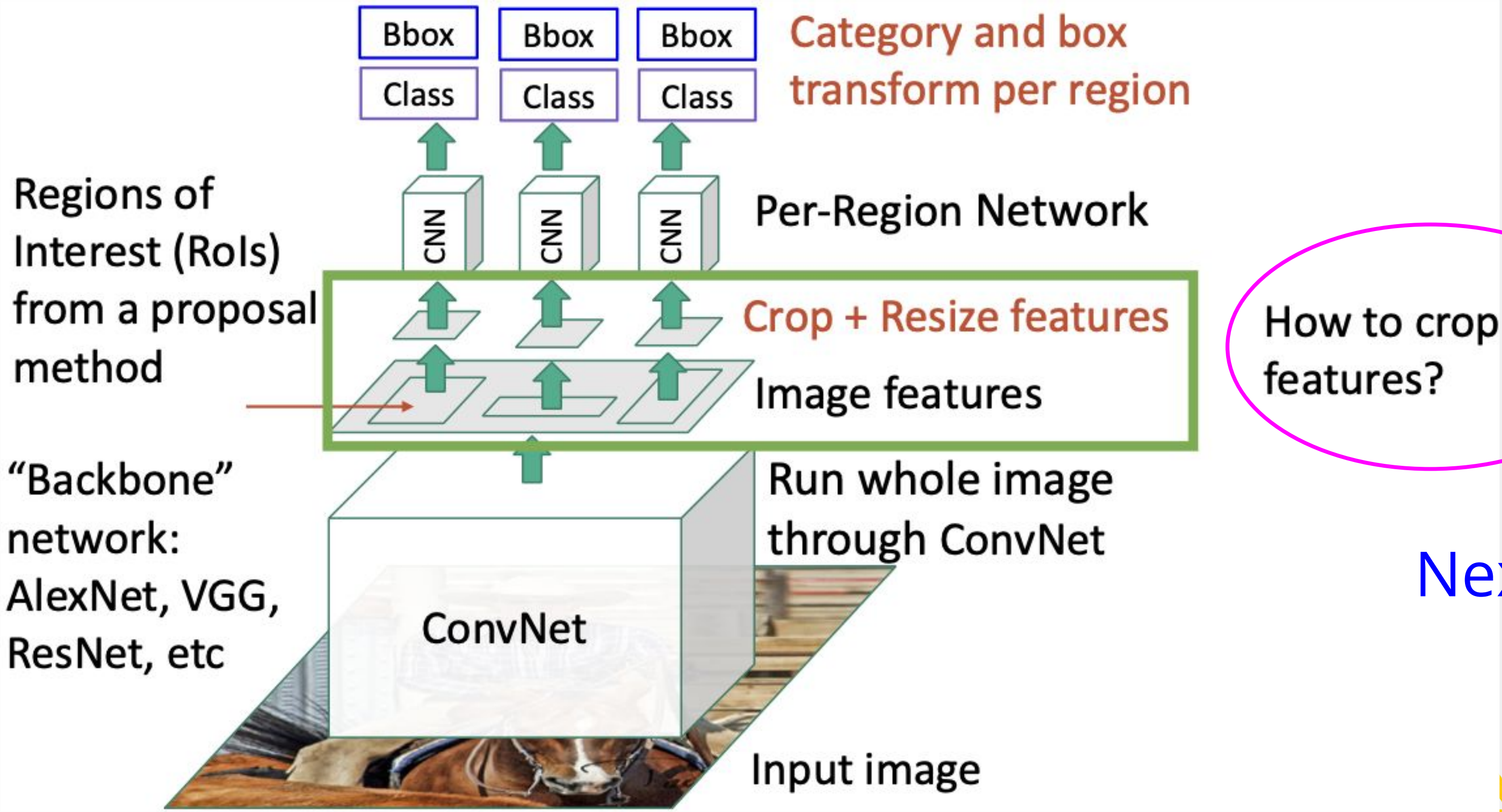
Example:
When using AlexNet for detection, five conv layers are used for backbone and two FC layers are used for per-region network

Fast R-CNN



Bbox
Class
Bbox
Class
Bbox
Class
Category and box transform per region
CNN
CNN
CNN
Per-Region Network
Regions of Interest (RoIs) from a proposal method
Crop + Resize features
Image features
"Backbone" network: AlexNet, VGG, ResNet, etc
ConvNet
Run whole image through ConvNet
Input image
Softmax
FC 1000
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512, /2
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128
3x3 conv, 128, /2
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
3x3 conv, 64
Pool
7x7 conv, 64, /2
Input
Example:
For ResNet, last stage is used as per-region network; the rest of the network is used as backbone

Fast R-CNN



How to crop features?

Next Time