

Object detection applied to the problem of robotic controlled pollination

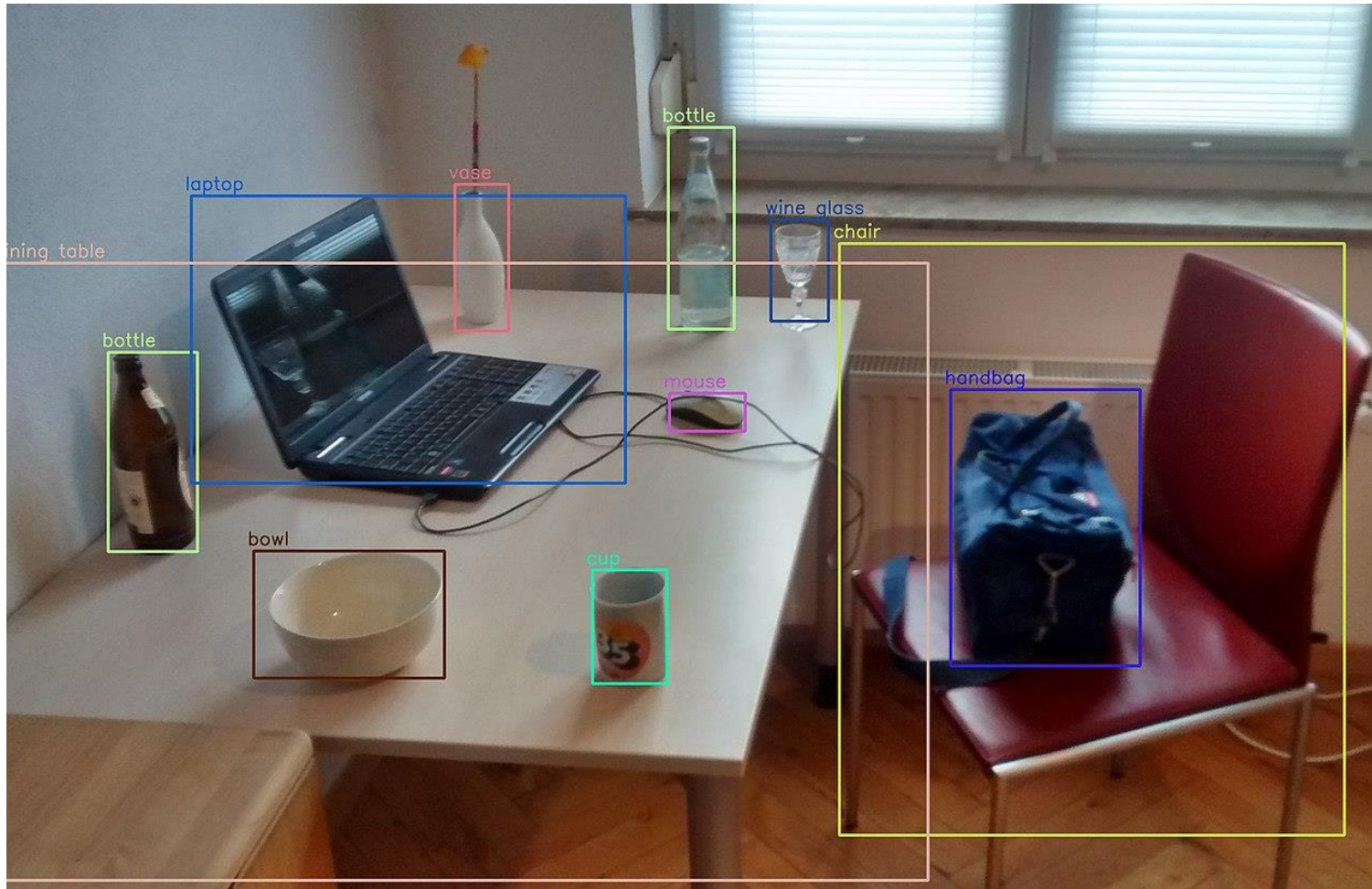
Piyush Pandey

2023/04/19

Overview

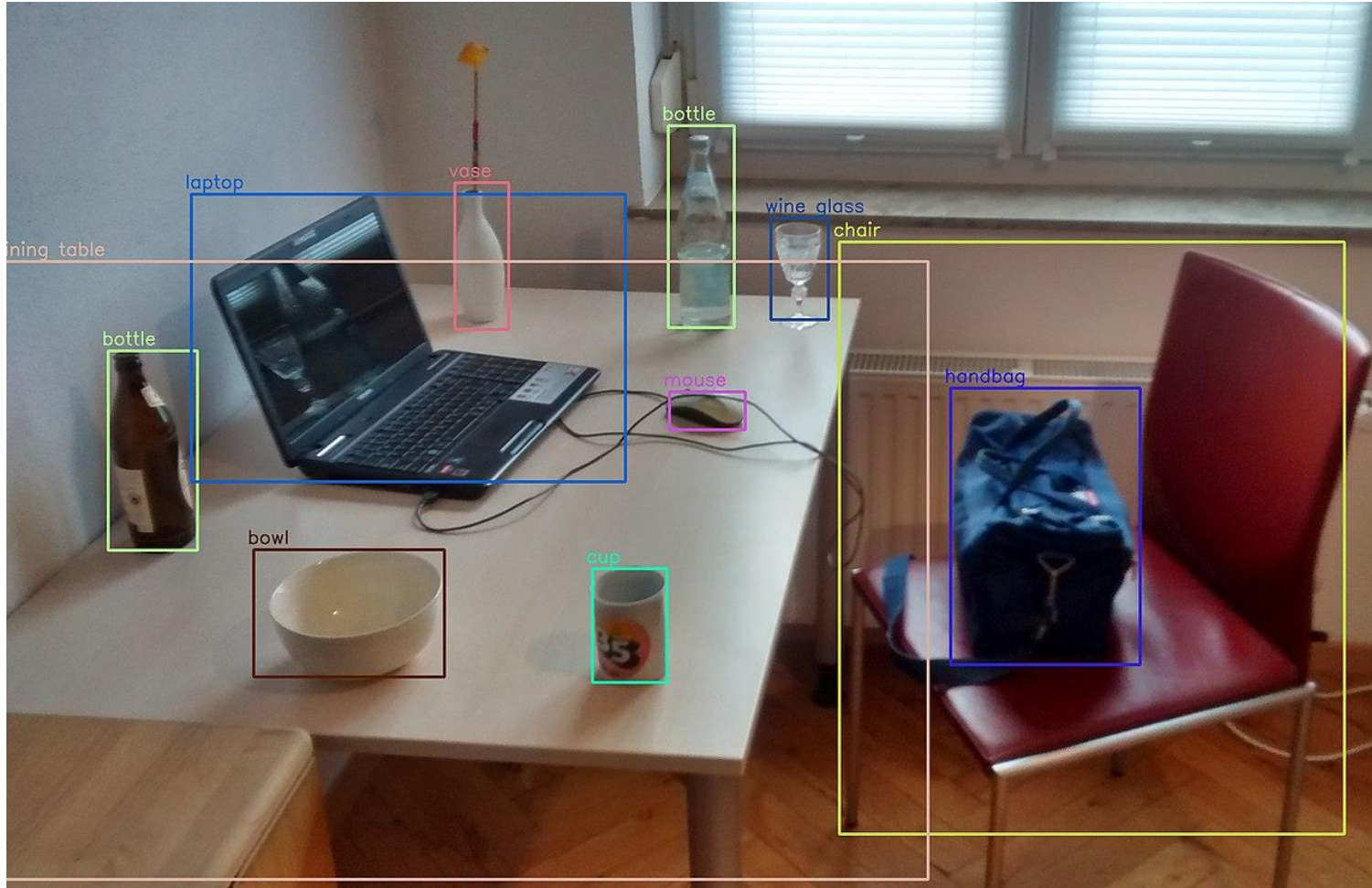
1. Object detection
2. Controlled pollination and the need for bag detection
3. Model architecture and workflow
4. Code implementation (Google Colab)

Object detection



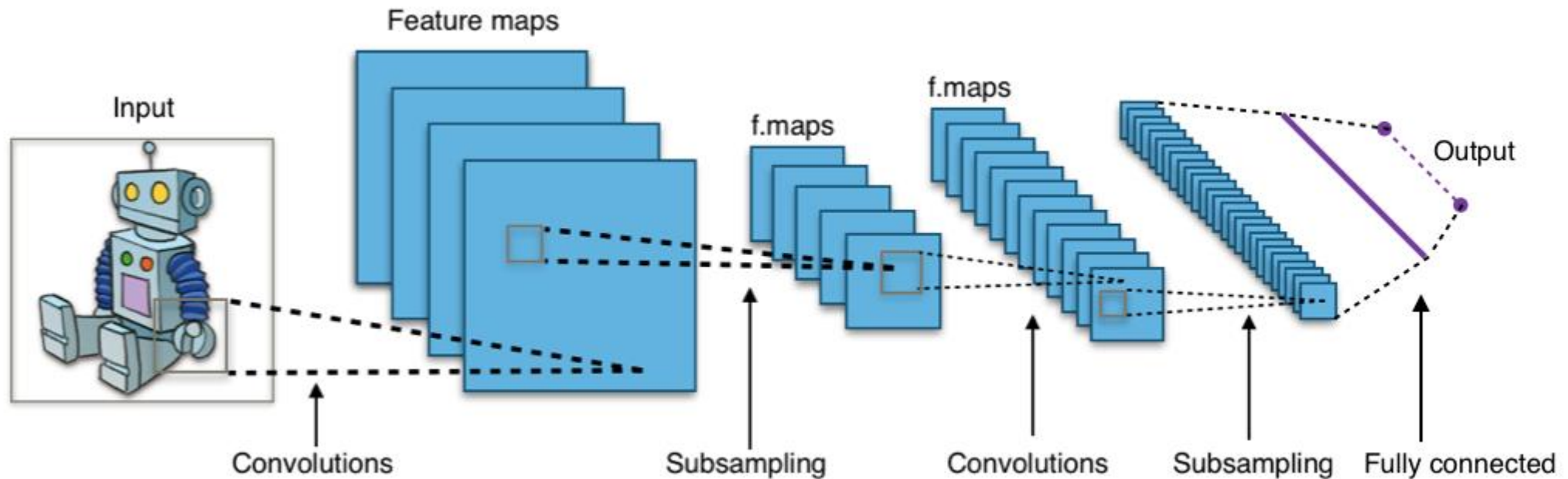
- Object detection includes the localization and classification of objects in the image
- Localization : bounding box
- Classification: box labels

Image classification



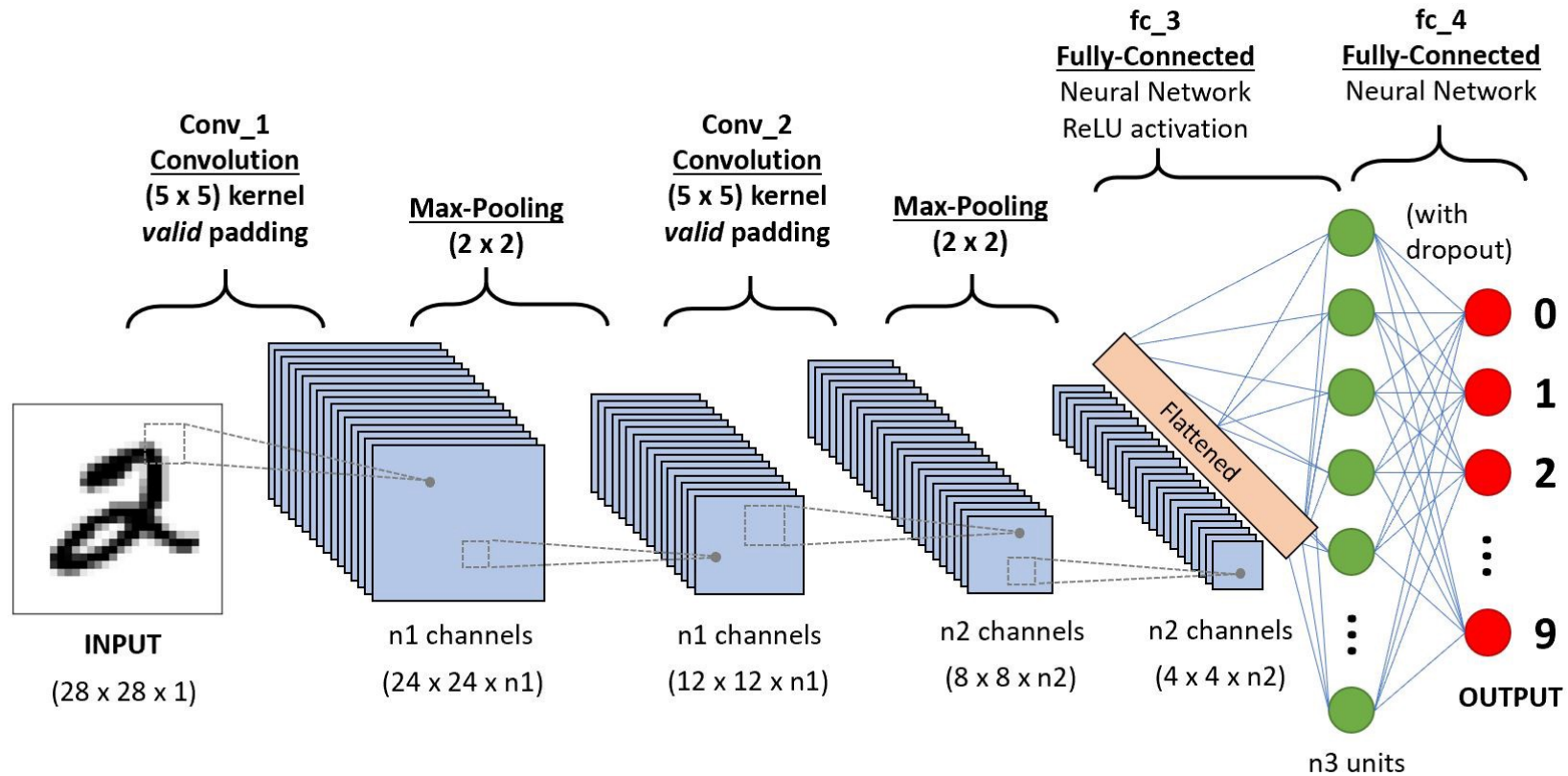
- Classification requires the assignment of a single label to the entire image
- A possible label for this image?

Image classification with CNNs



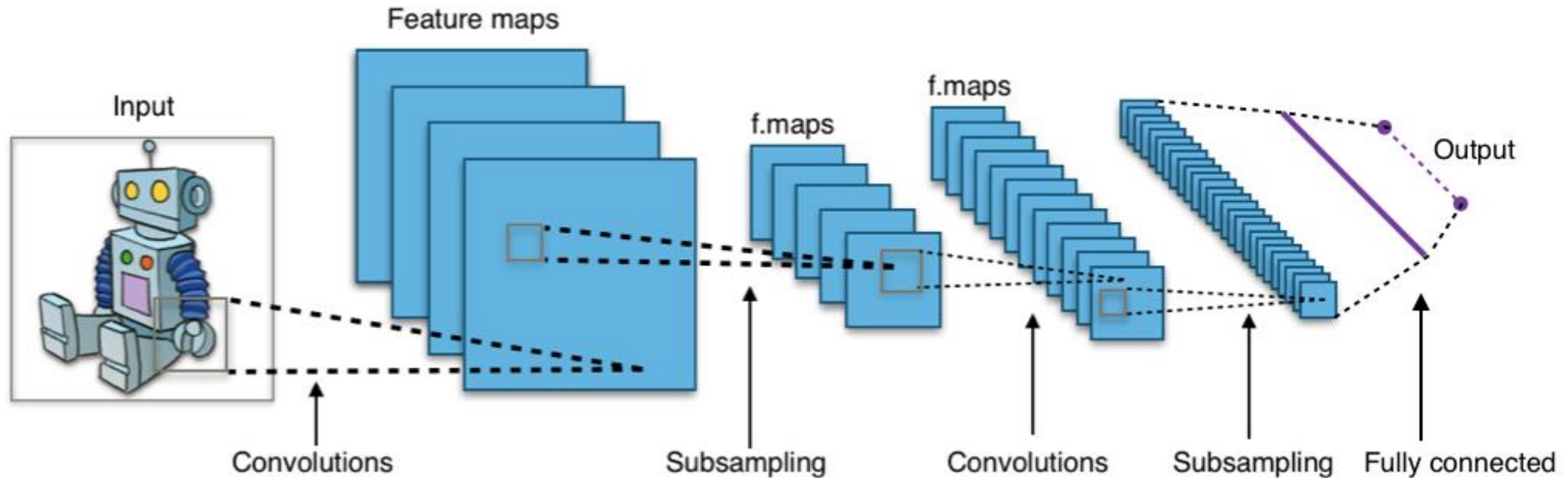
Typical CNN model with convolutional and subsampling (pooling) layers followed by fully connected layers

Image classification with CNNs



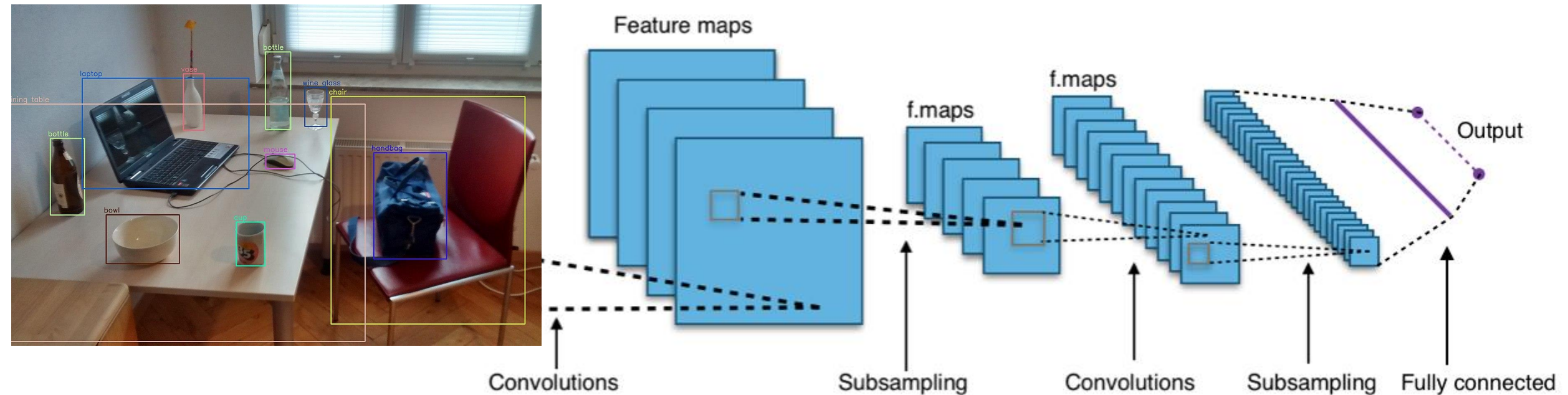
Typical CNN model with convolutional and subsampling (pooling) layers followed by fully connected layers

Object detection with CNNs



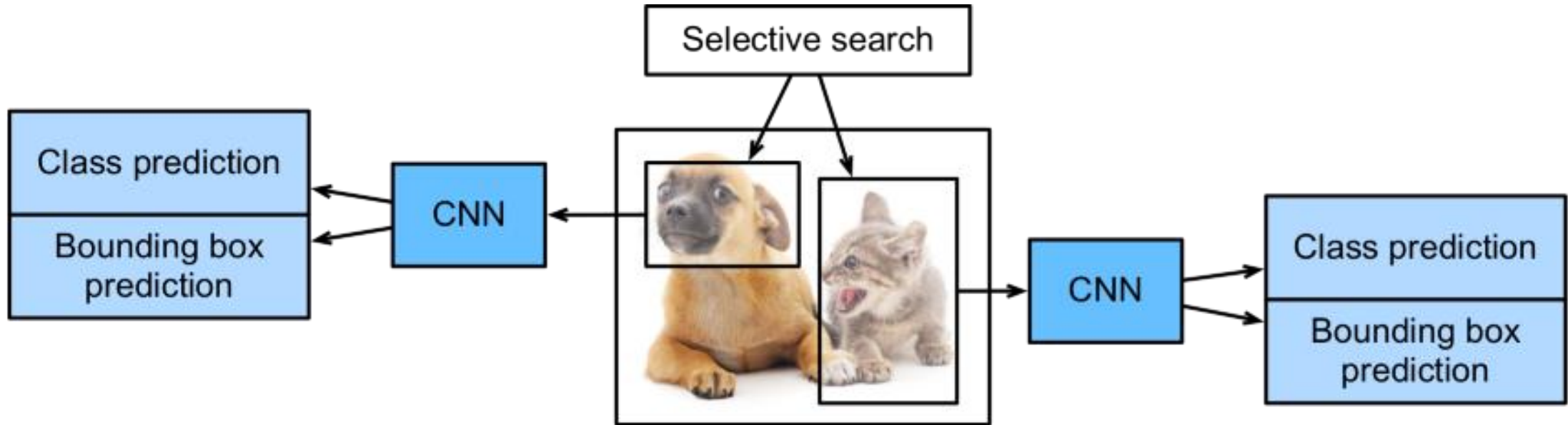
Object detection can be understood as the problem of operating this typical CNN model on an image patch

Object detection with CNNs

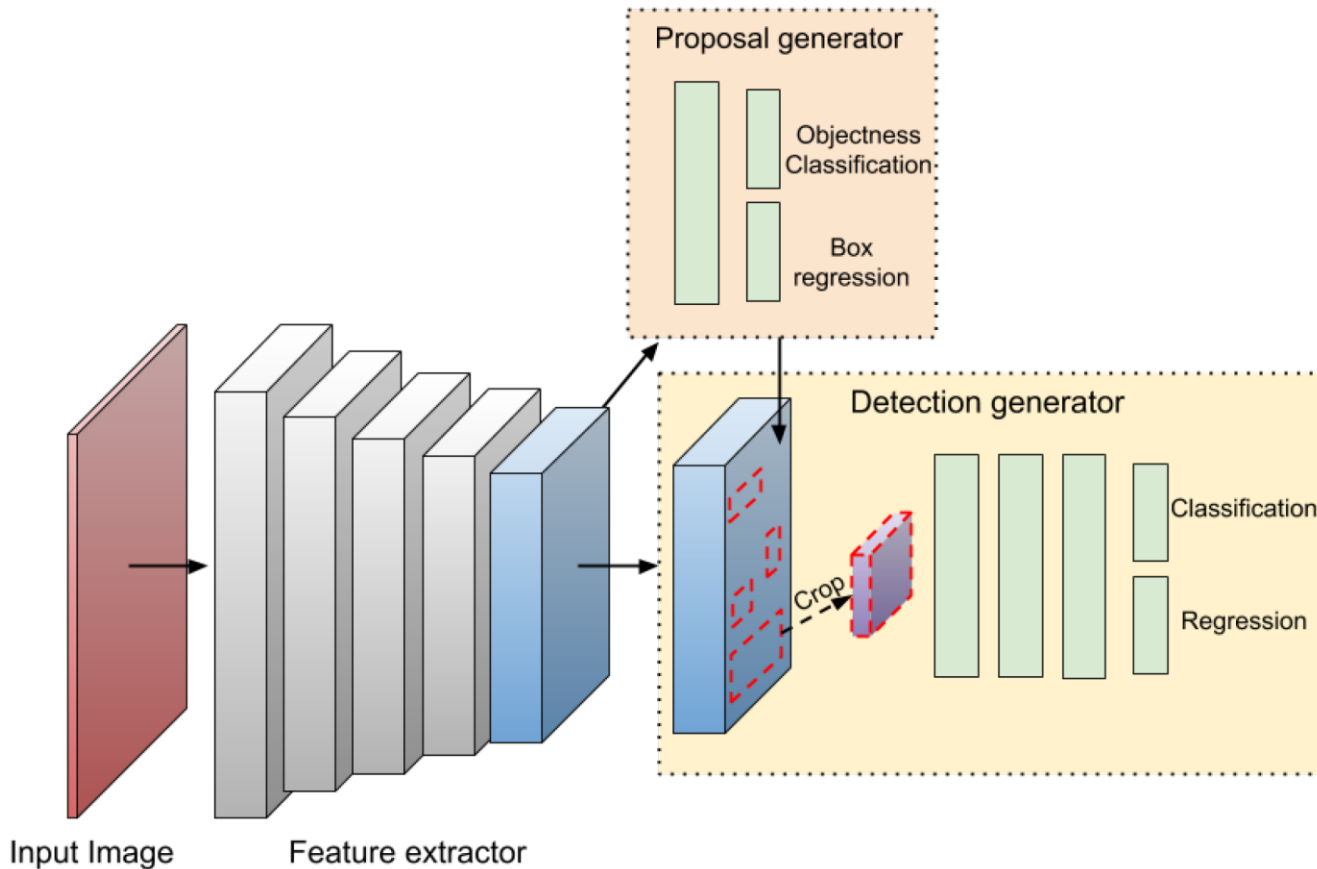


Object detection can be understood as the problem of operating this typical CNN model on an image patch

Object detection with CNNs



Two-stage object detectors



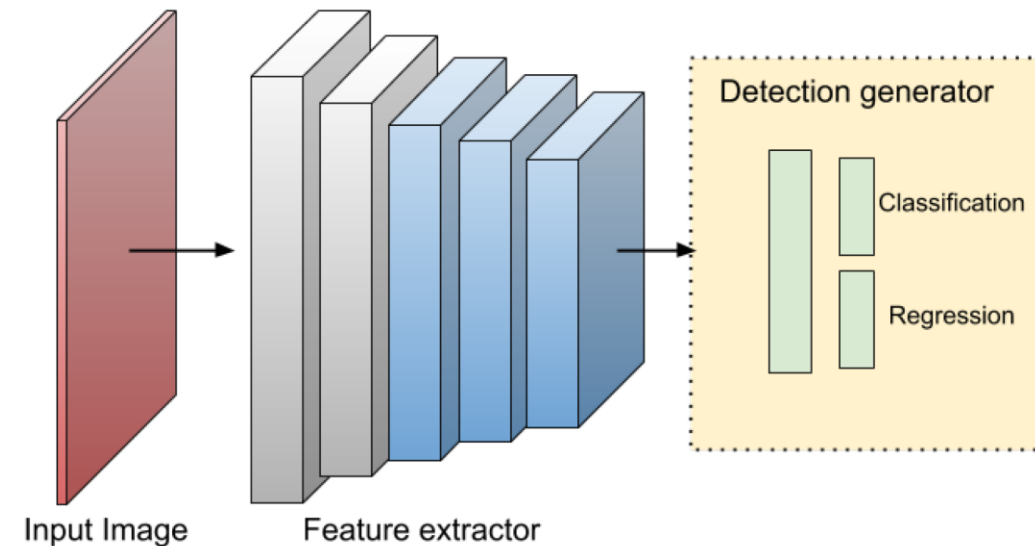
(a) Two-stage Faster R-CNN

In a two-stage object detector, possible patches of object location are first identified followed by the classification of these patches and the refinement of the location

Two-stage object detectors

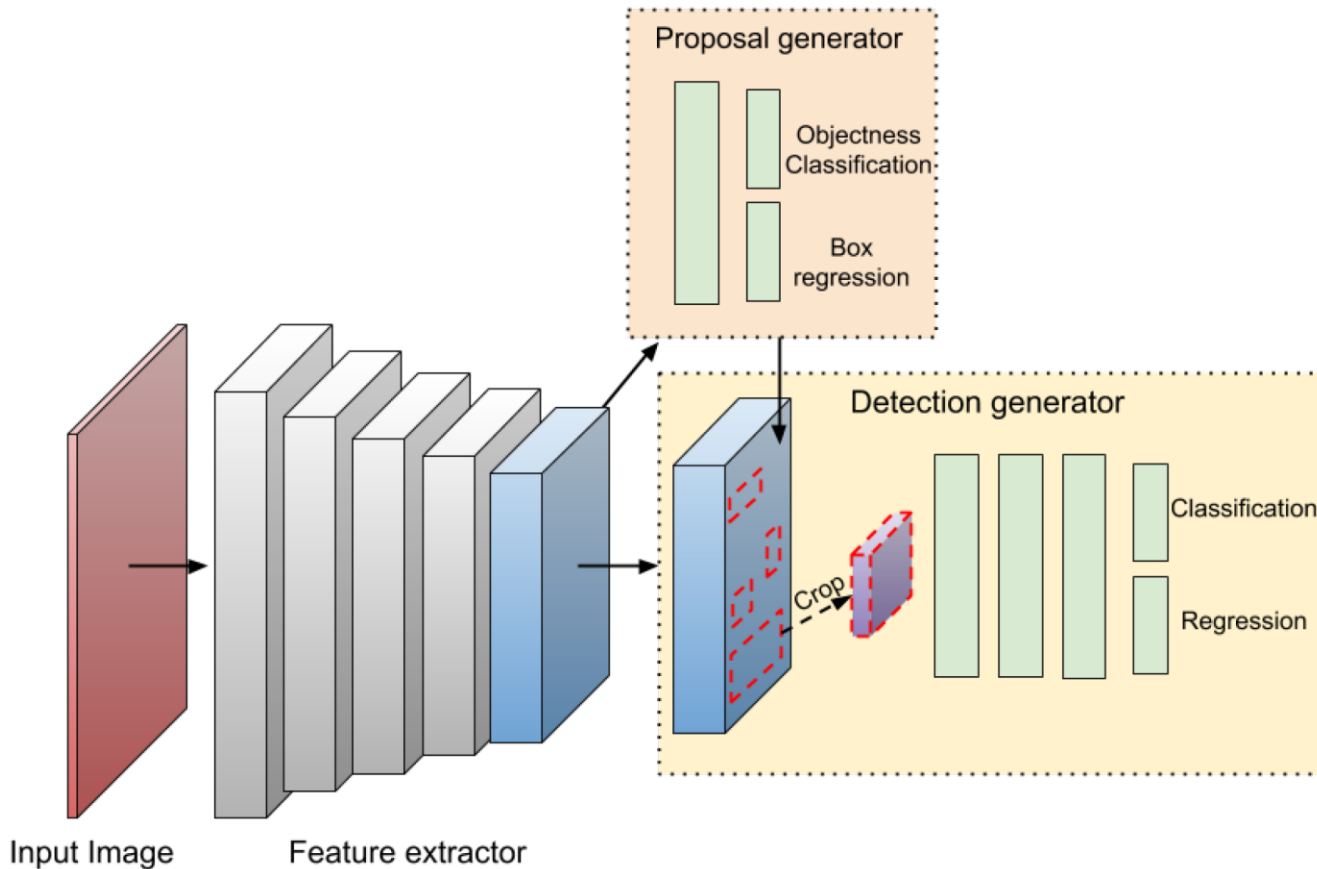
In a one-stage object detection model, bounding box location and label are simultaneously predicted by the model

The recognition of the patch with possible object is included in the model architecture

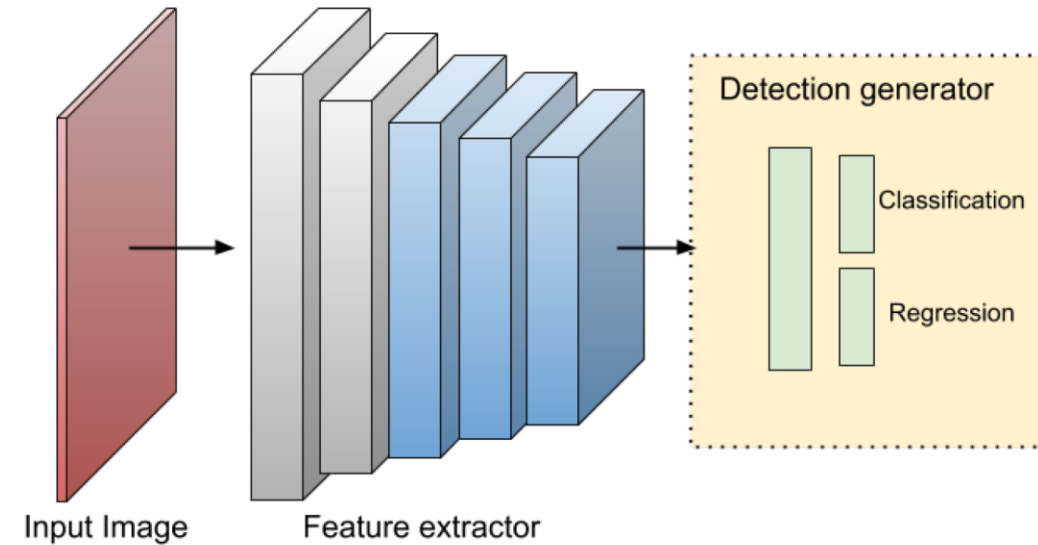


(b) One-stage RetinaNet

Two-stage object detectors



(a) Two-stage Faster R-CNN



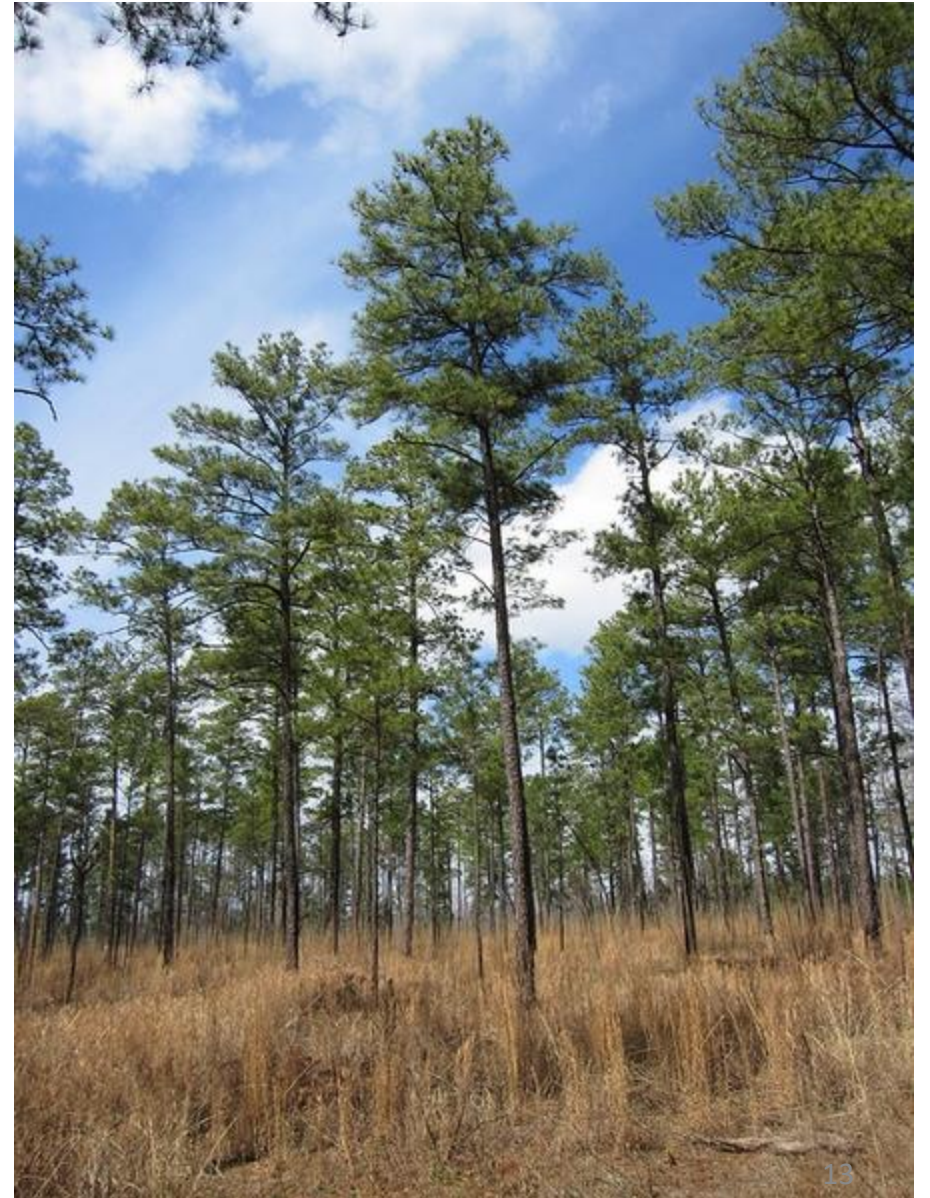
(b) One-stage RetinaNet

Carranza-García, M., Torres-Mateo, J., Lara-Benítez, P., & García-Gutiérrez, J. (2020). On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data. *Remote Sensing*, 13(1), 89.

Robotic controlled pollination in loblolly pine

- Timber tree used for pulp, plywood, general construction
- Dominant tree species in NC and the southeast
- In 2019, forest sector generated \$21.6 billion in industry output¹

¹ Parajuli, R.et al. (2020). *Economic Contribution of the Forest Sector in North Carolina, 2019*. NC State Extension, NC State University, College of Natural Resources.



Robotic controlled pollination in loblolly pine



Robotic controlled pollination in loblolly pine



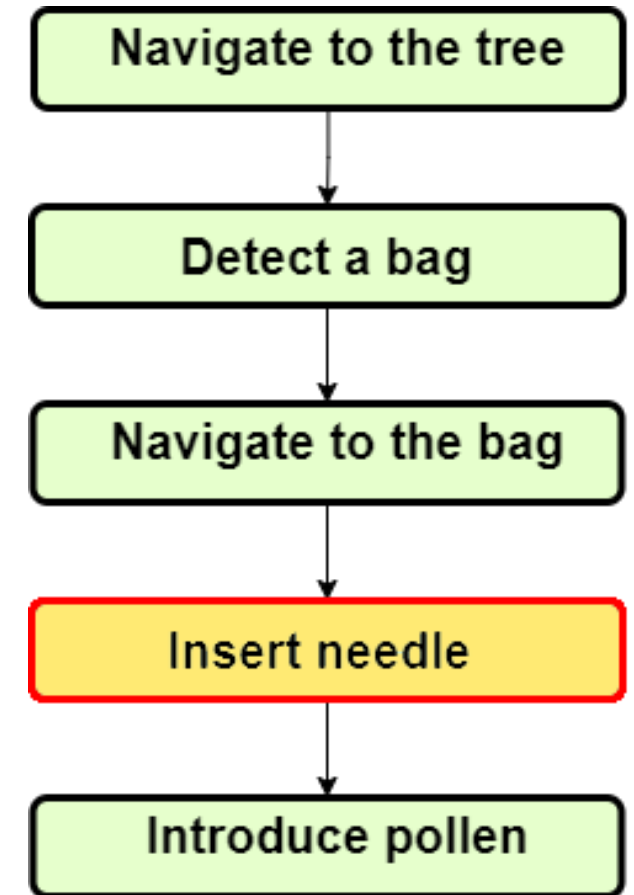
Pneumatic injectors 15

Robotic controlled pollination in loblolly pine



- More than a million bags placed annually
- Narrow time-window for pollination
- Challenges due to weather
- Safety
- Soil compaction

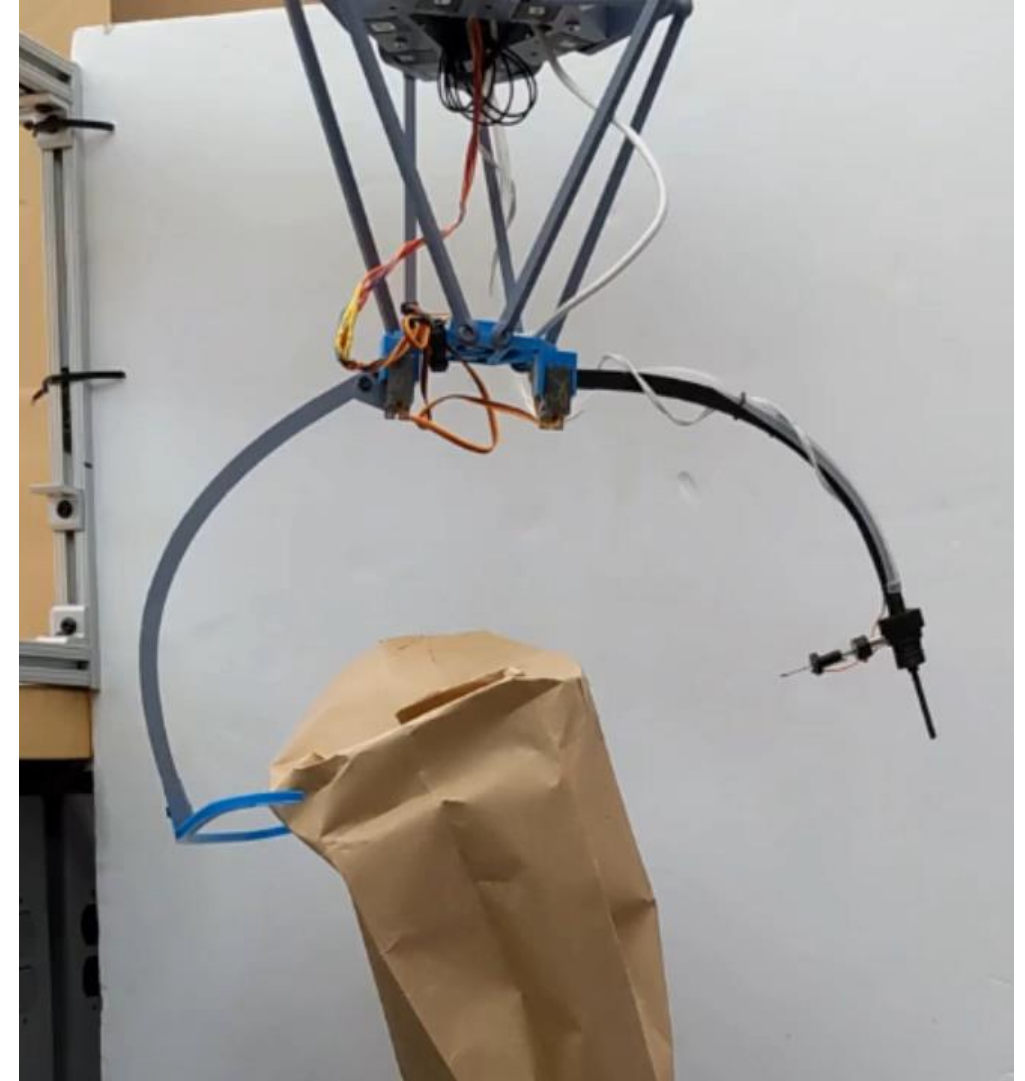
Robotic controlled pollination in loblolly pine



Robotic controlled pollination in loblolly pine



DUO MLX
Stereo camera



Robotic controlled pollination in loblolly pine



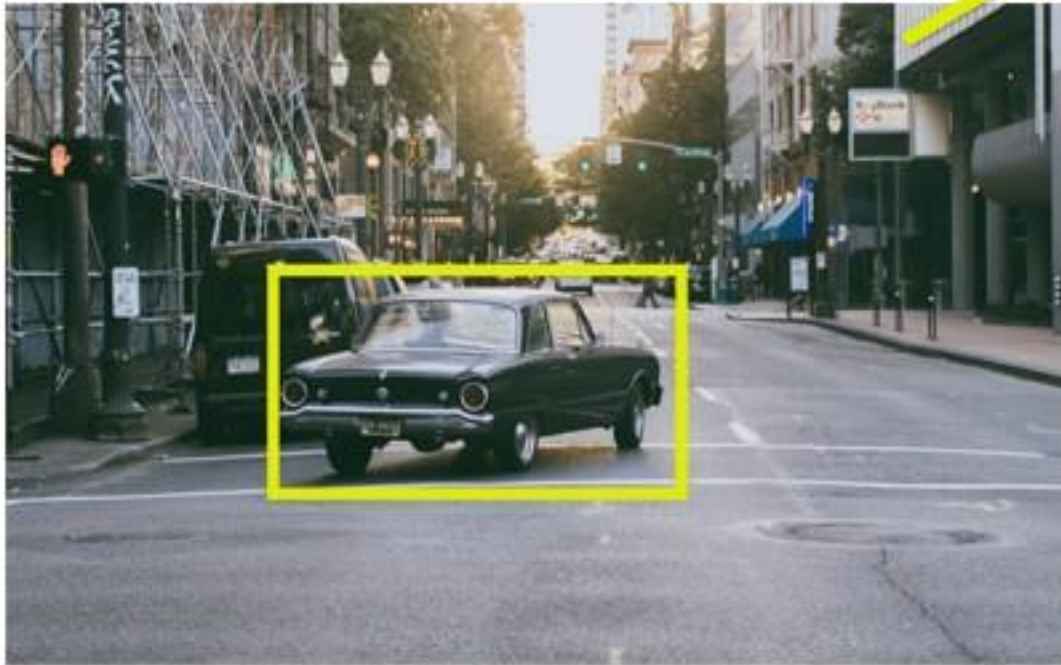
DUO MLX
Stereo camera

Classification or detection?

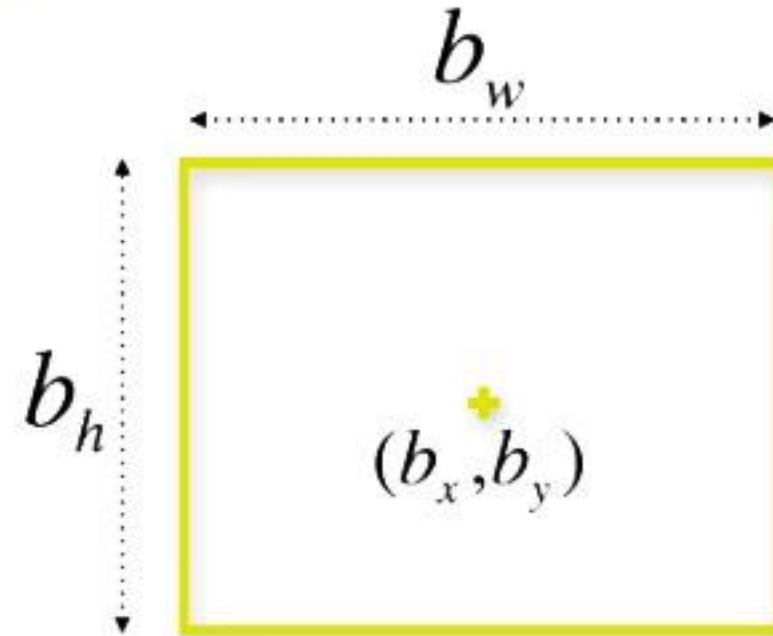
One-stage object detection model or a two-stage model?

What about segmentation?

You Only Look Once (YOLO)



$$y = (p_c, b_x, b_y, b_h, b_w, c)$$

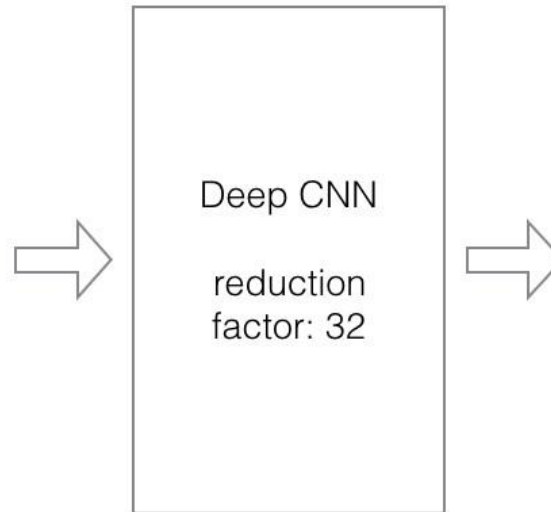


<https://appsilon.com/object-detection-yolo-algorithm/>

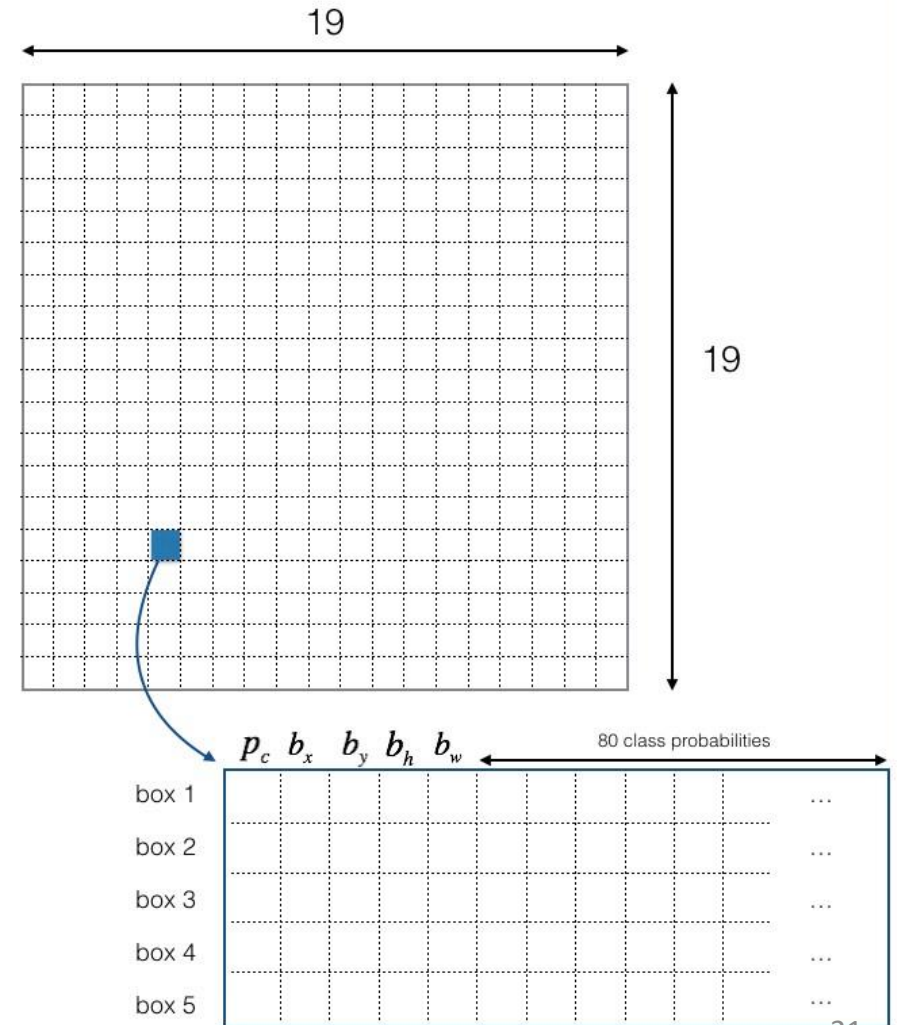
We predict bounding box location (b_x, b_y, b_h, b_w) , probability of objectness (p_c) and the object class label (c) .

You Only Look Once (YOLO)

preprocessed image
(608, 608, 3)



encoding
(19, 19, 5, 85)



Instead of finding “possible object locations”,
we process all locations

<https://appsilon.com/object-detection-yolo-algorithm/>

YOLO Idea 1: NMS

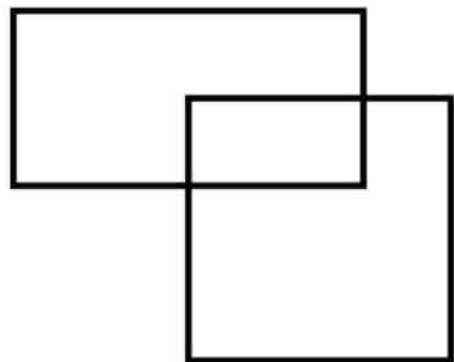


Applying
→
NMS

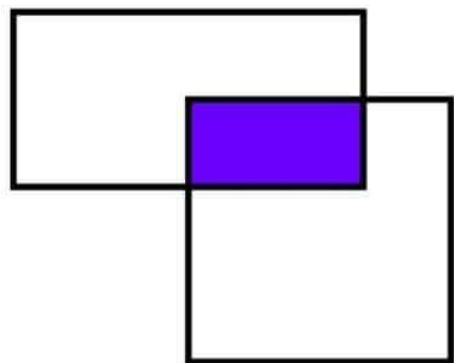


[LearnOpenCV.com](https://learnopencv.com)

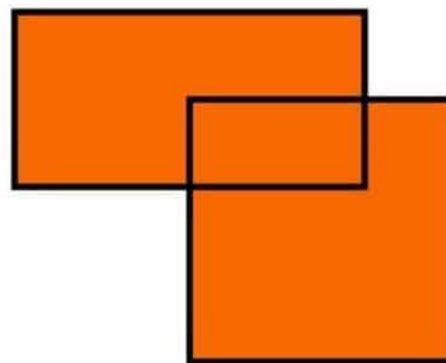
YOLO Idea 2: IoU



**For a set of bounding boxes
like the given one**



The purple area is Intersection

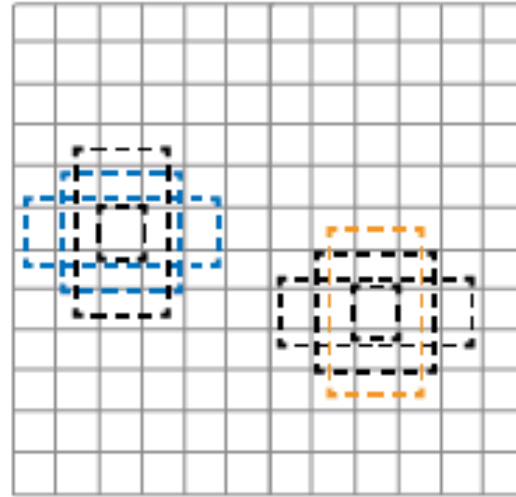


The orange area is Union

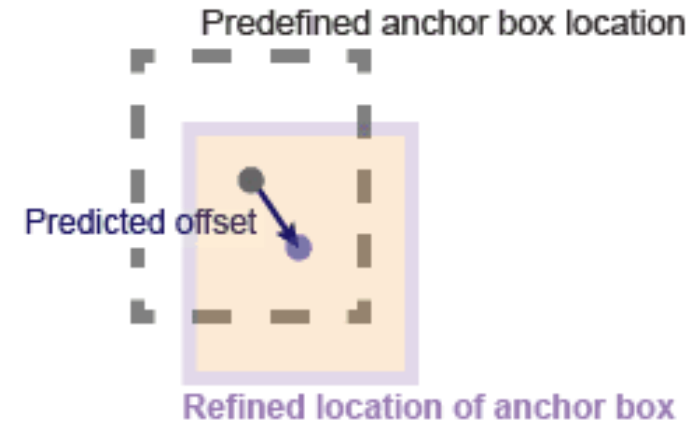
YOLO Idea 3: Anchor boxes



Ground truth image and bounding boxes



Anchor boxes at each predefined location in each feature map



Two anchor boxes
Class: airplane
Class: sailboat



Filter by class scores,
perform non-max suppression
and intersection over union



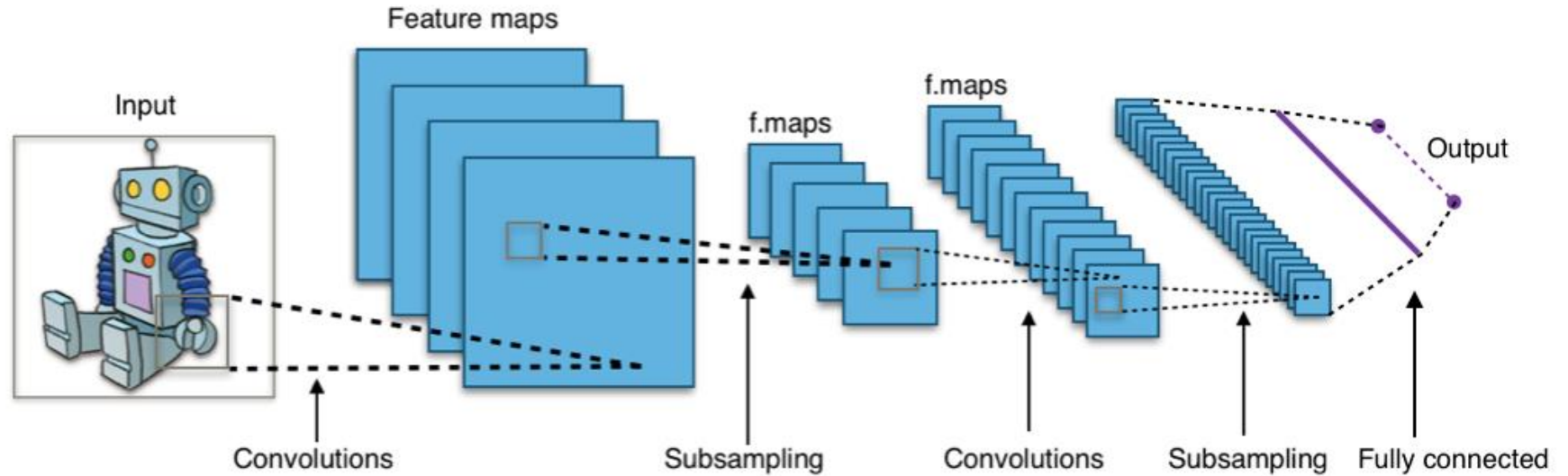
Model training pipeline

1. Select model architecture
2. Decide training process (Training from scratch? Transfer learning?)
3. Label images
4. Split data into train-validation-testing
5. Train the model and iterate

Transfer learning

The initial layers are frozen

Existing model architecture, pre-trained on large dataset...



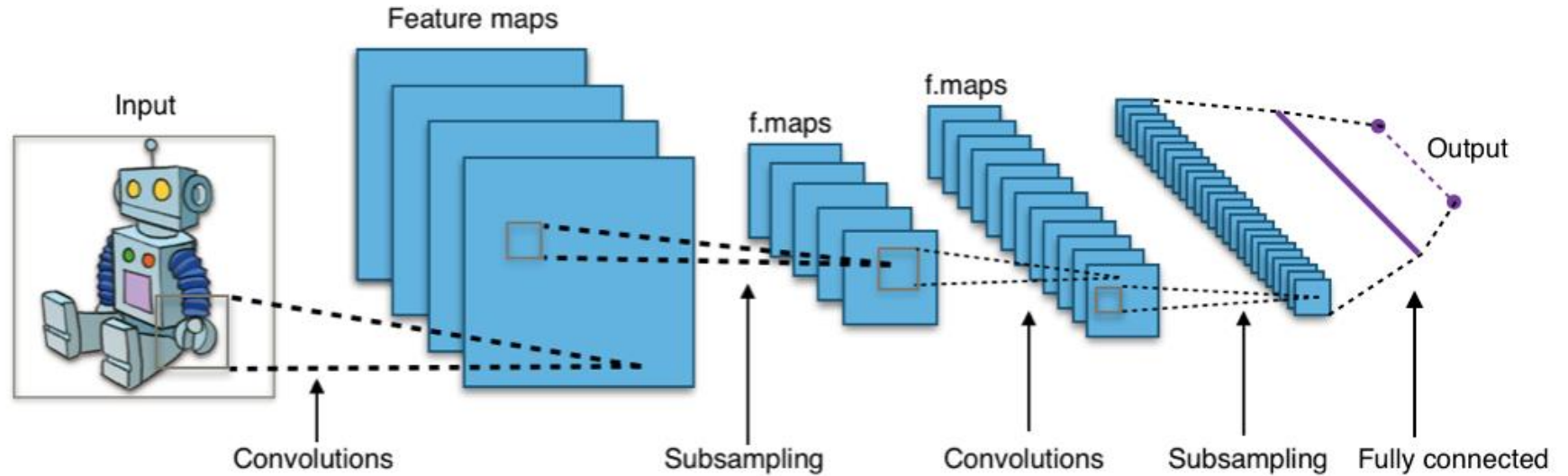
But not trained on paper bags put on tree branches.

Freezing parts of the pretrained model (keeping the trained parameters unchanged) leads to faster training on the novel dataset.

Transfer learning

The initial layers are frozen

Existing model architecture, pre-trained on large dataset...



But not trained on paper bags put on tree branches.

Freezing parts of the pretrained model (keeping the trained parameters unchanged) leads to faster training on the novel dataset.

Model training pipeline

1. Select model architecture: **one-stage detector (YOLOv5)**
2. Decide training process: **Transfer learning**
3. Label images
4. Split data into train-validation-testing
5. Train the model and iterate

Labeling tools

Labeling for object detection is simply drawing a rectangle and assigning a class label

1. MATLAB image labeler: Computer vision toolbox

Stable, intuitive, and reliable, not free

2. LabelImg: <https://github.com/heartexlabs/labelImg> (Transitioned to a new tool (Label Studio))

Intuitive and free, limited functionality

3. VGG Image Annotator: <https://www.robots.ox.ac.uk/~vgg/software/via/>

Simple and free but not user friendly

Larger/ advanced applications: CVAT, LabelStudio

FILE VIEW LABEL OPACITY AUTOMATE LABELING RESOURCES SUMMARY LAYOUT EXPORT

Label

New Session Open Session Save Session Import

Show ROI Labels ROI Color On Hover By Label

Label Opacity Polygon Pixel

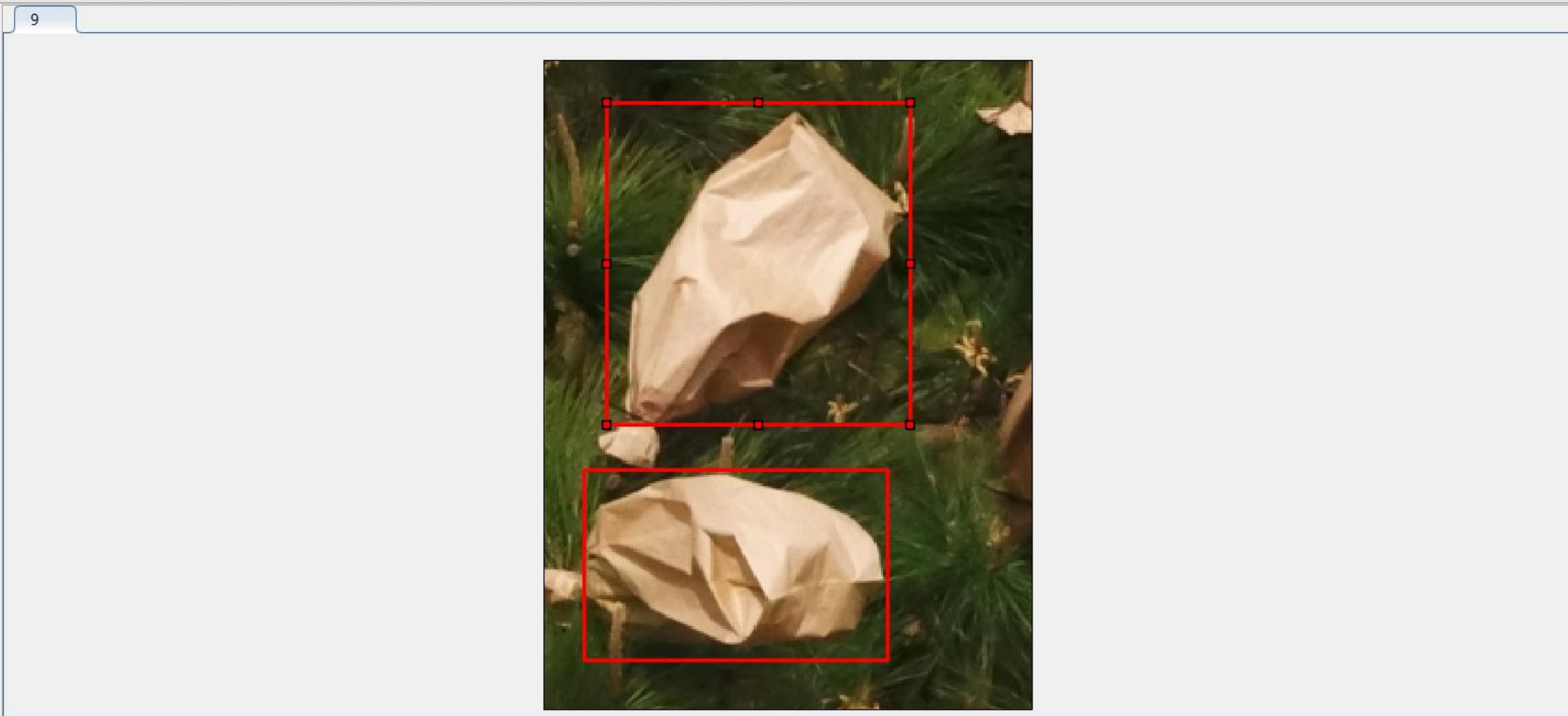
Algorithm: Select Algorithm Automate

View Shortcuts View Label Summary Layout Export

ROI Labels Scene Labels

Label Sublabel Attribute

bag



Open

Open Dir

Change Save Dir

Next Image

Prev Image

Verify Image

Save

yolo

YOLO

Create RectBox

Duplicate RectBox

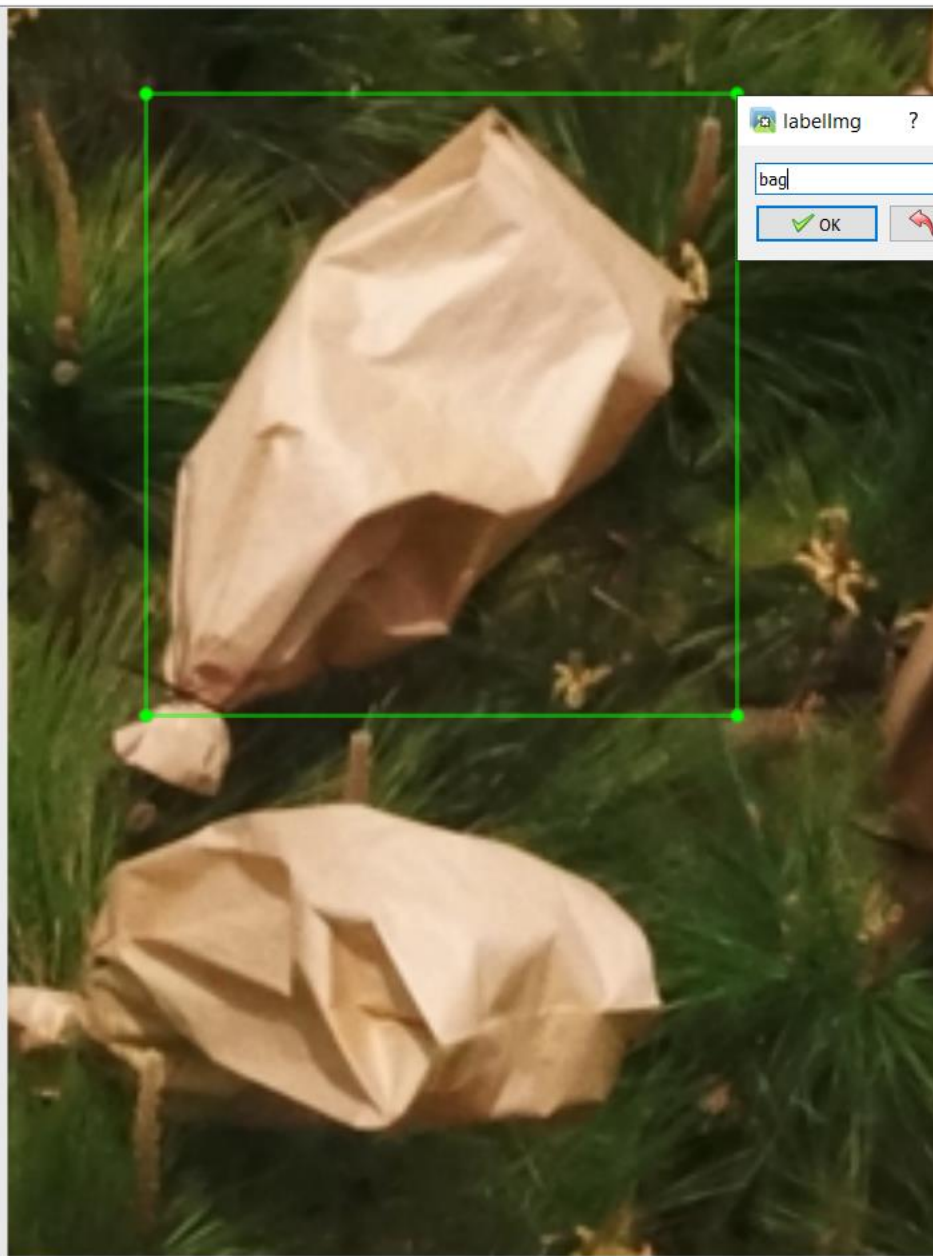
Delete RectBox

Zoom In

326 %

Zoom Out

Fit Window



labellmg ? X

bag

OK Cancel

Box Labels

Edit Label

☐ difficult


☐ Use default label

File List

Width: 136, Height: 143 / X: 167; Y: 19

Model training pipeline

1. Select model architecture: **one-stage detector (YOLOv5)**
2. Decide training process: **Transfer learning**
3. Label images
4. Split data into train-validation-testing
5. Train the model and iterate

For steps 4 and 5, visit <https://github.com/piyuss/bag-detection>, click on Exclusion_bag_detection_demo_YOLOv5.ipynb, click on  Open in Colab. You will need a Google account and will need to log in to your account to run the code.