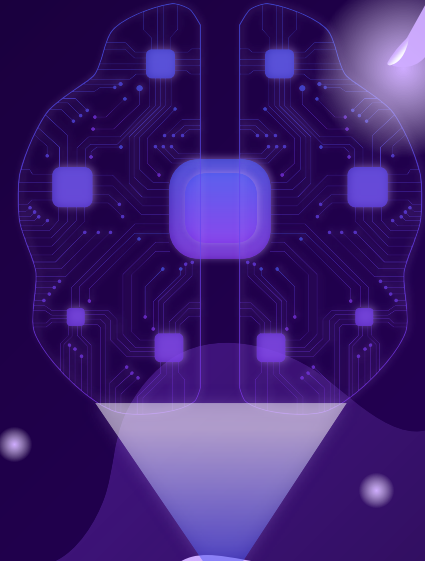


# Object Detection 101



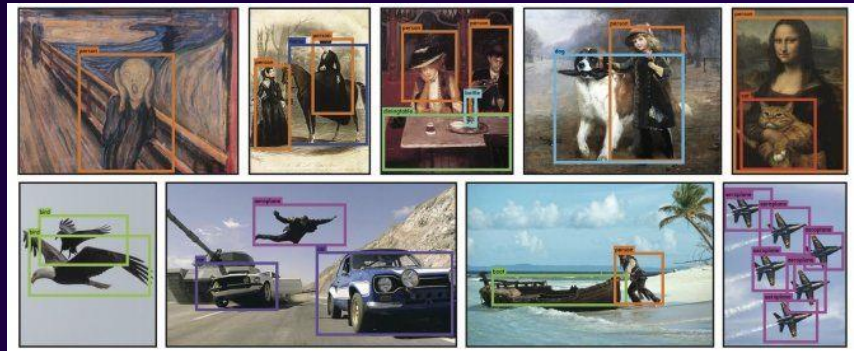
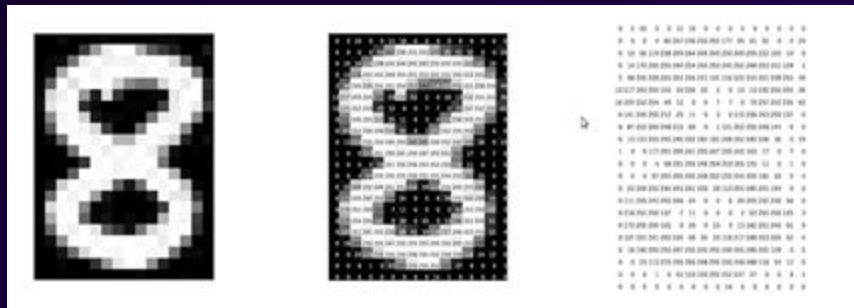
# Welcome!

Today, we will show you guys how to use  
one key AI tools:

- YOLOv8
- YOLOv8-obb




# Computer Vision???

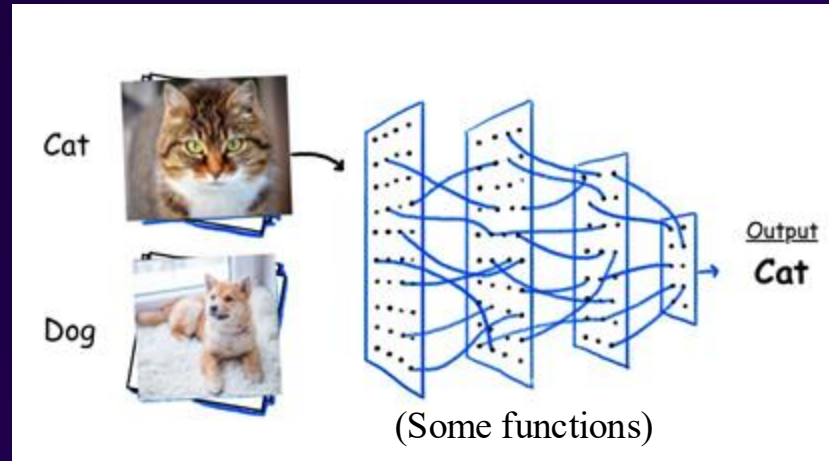
- How can computers see?
- **Low-level:** colors, intensity difference
- **High-level:** features, semantics



# Main idea

- Give computer an **image** and a **label**. The label tells the computer what is in the picture.
- Computer **converts** the image to **numbers** and find **pattern**.
- Give computer a new image **without a label**. Computer predicts based on the pattern it learned before.
- If computer is correct, **keep** the pattern
- If it's wrong, **adjust** the pattern.
- This is repeated a looooooot of times, until computer is corrected and makes accurate guesses

Data	Label
	A cat
	Not a cat
	A cat

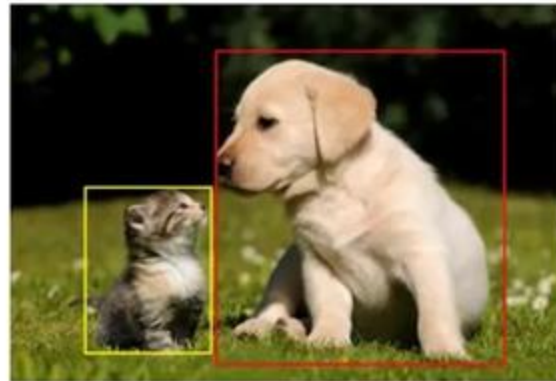


Is this a dog?



Image Classification

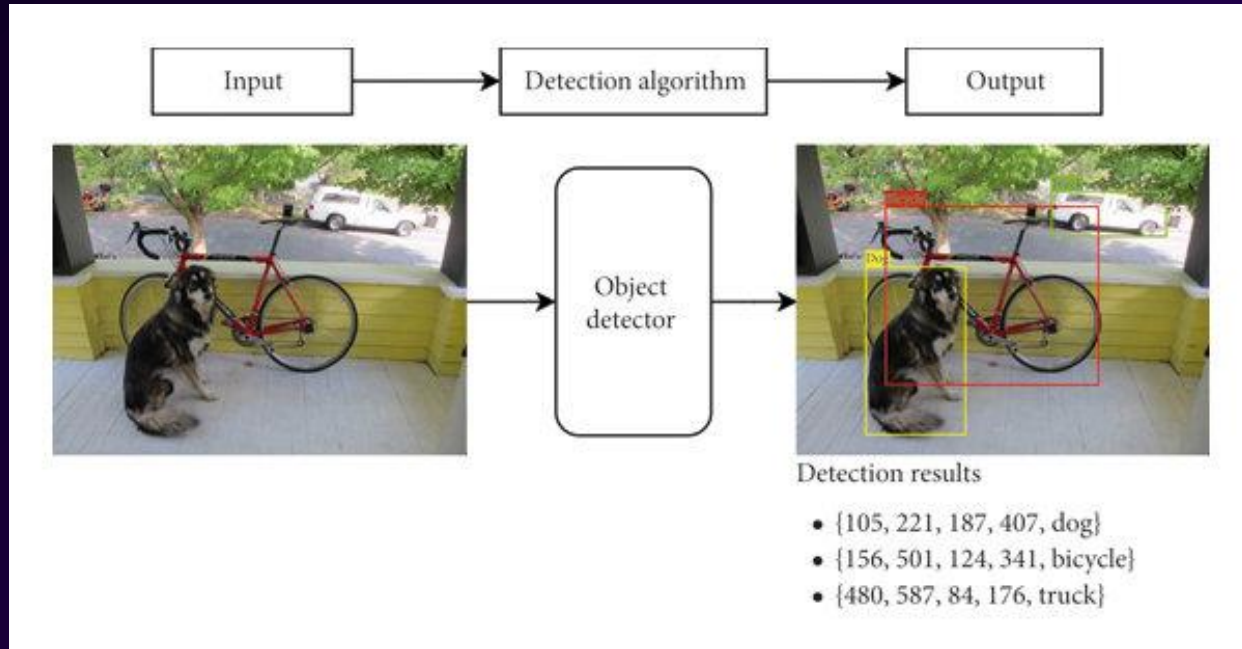
What is there in image  
and where?



Object Detection

# Object Detection...

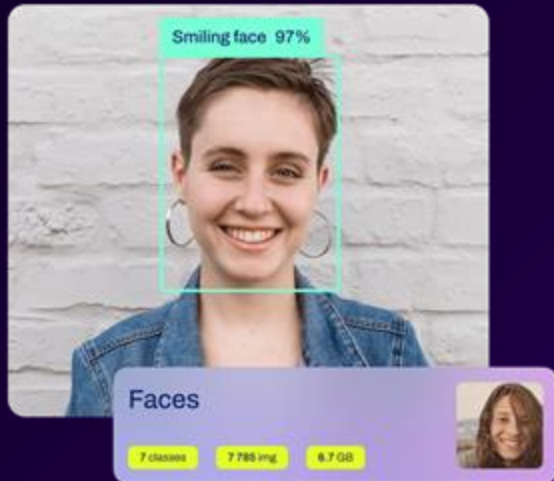
The training works pretty much the same, but the algorithm has a few more things to calculate:





# YOLOv8??

- VERY famous machine learning algorithm
- Intended to be used in REAL-TIME by doing the important calculations in a single step
- Supported by standard hardware (nothing too fancy needed)



# How does it work?

NOTE: this is an overly simplified explanation.

The picture is divided into cells. For each cell, YOLOv8 guesses where objects are present (bounding boxes) and also guesses what each object is (class probabilities)

Then, YOLOv8 selects only the most likely one for each object in the picture.





Let us get our hands dirty with Yolov8! 🤖 🤖

# STEP-BY-STEP SET up Environment

1. Run:

`git clone https://github.com/HKUArmStrong/rac\_workshop`

2. Setup Python environment:

`conda create --name cv_workshop python=3.12`

`conda activate cv_workshop`

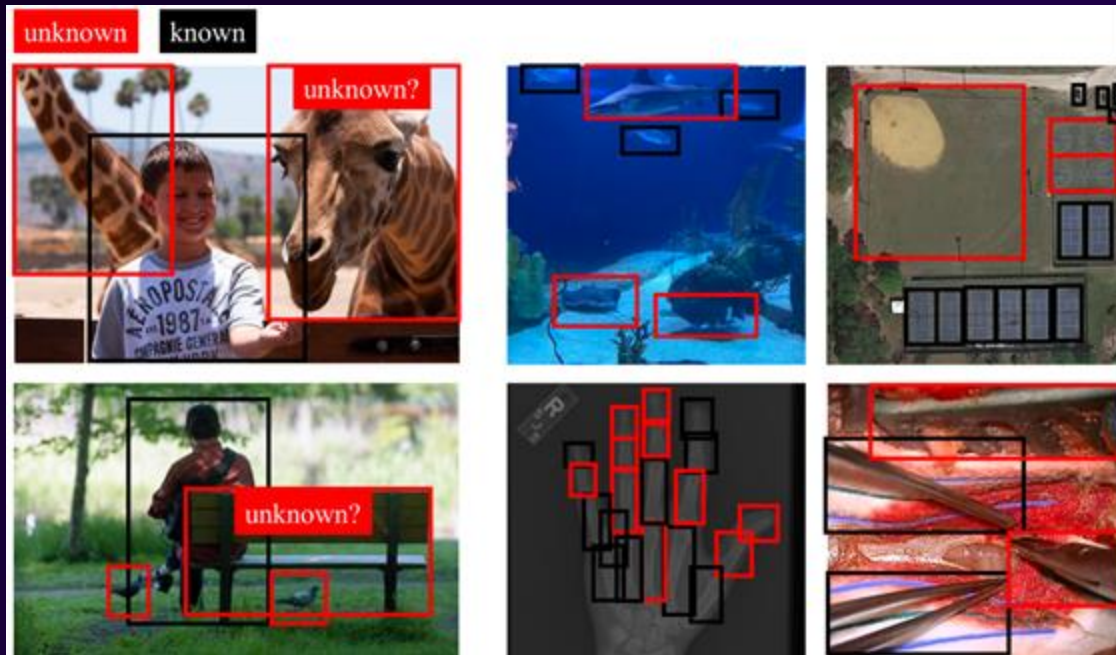
3. With requirements.txt:

`pip install -r requirements.txt`

4. Switch to different `.txt` file at step 3 to be compatible to your system. (E.g., on Mac / with GPU / CPU only)

## The only things Yolo can detect:

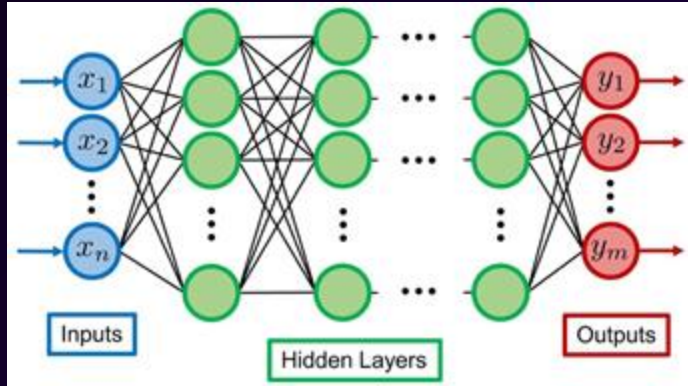
{0: 'person', 1: 'bicycle', 2: 'car', 3: 'motorcycle', 4: 'airplane', 5: 'bus', 6: 'train', 7: 'truck', 8: 'boat', 9: 'traffic light', 10: 'fire hydrant', 11: 'stop sign', 12: 'parking meter', 13: 'bench', 14: 'bird', 15: 'cat', 16: 'dog', 17: 'horse', 18: 'sheep', 19: 'cow', 20: 'elephant', 21: 'bear', 22: 'zebra', 23: 'giraffe', 24: 'backpack', 25: 'umbrella', 26: 'handbag', 27: 'tie', 28: 'suitcase', 29: 'frisbee', 30: 'skis', 31: 'snowboard', 32: 'sports ball', 33: 'kite', 34: 'baseball bat', 35: 'baseball glove', 36: 'skateboard', 37: 'surfboard', 38: 'tennis racket', 39: 'bottle', 40: 'wine glass', 41: 'cup', 42: 'fork', 43: 'knife', 44: 'spoon', 45: 'bowl', 46: 'banana', 47: 'apple', 48: 'sandwich', 49: 'orange', 50: 'broccoli', 51: 'carrot', 52: 'hot dog', 53: 'pizza', 54: 'donut', 55: 'cake', 56: 'chair', 57: 'couch', 58: 'potted plant', 59: 'bed', 60: 'dining table', 61: 'toilet', 62: 'tv', 63: 'laptop', 64: 'mouse', 65: 'remote', 66: 'keyboard', 67: 'cell phone', 68: 'microwave', 69: 'oven', 70: 'toaster', 71: 'sink', 72: 'refrigerator', 73: 'book', 74: 'clock', 75: 'vase', 76: 'scissors', 77: 'teddy bear', 78: 'hair drier', 79: 'toothbrush'}



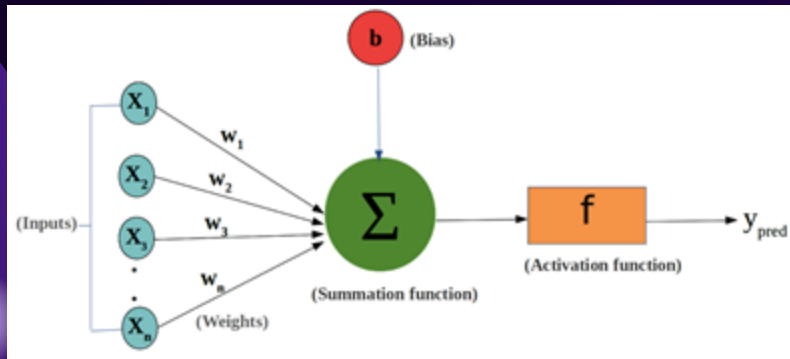
If you want to detect something else, **YOU MUST TRAIN THE MODEL!!!**

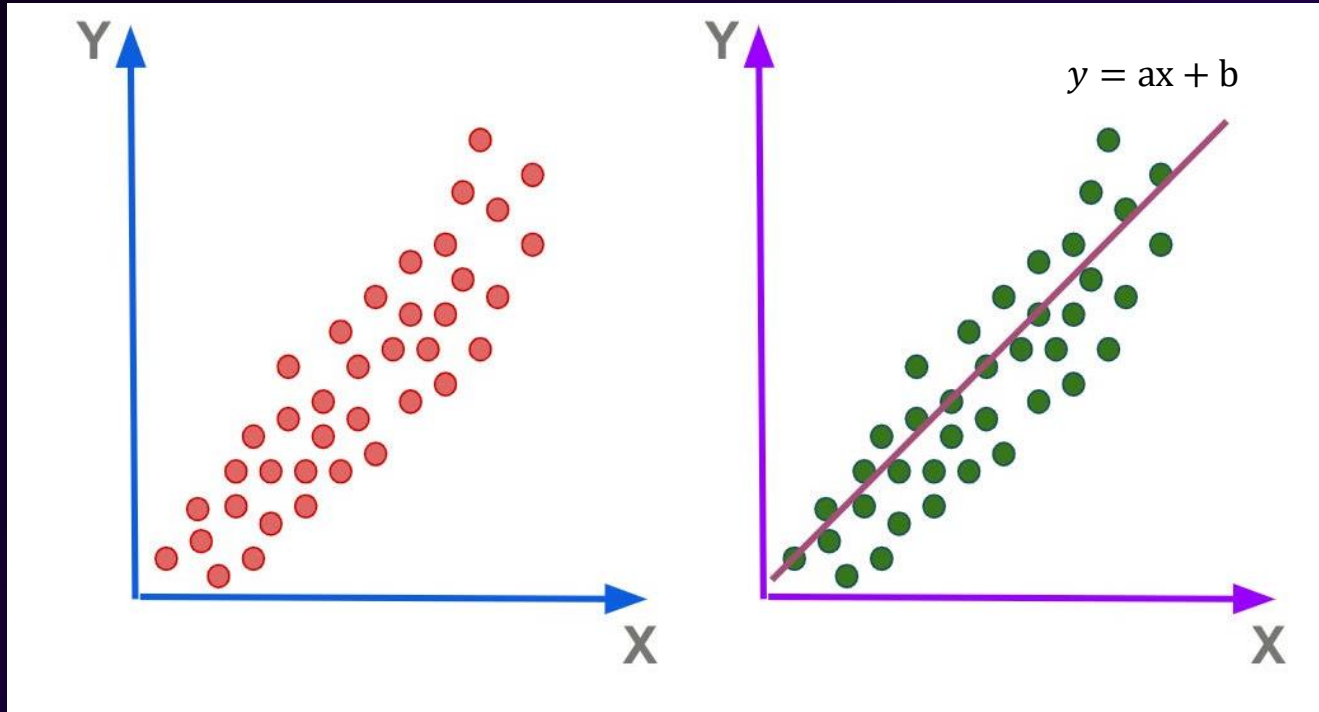
This is called **“closed-set object detection”**...

# How to train a model?

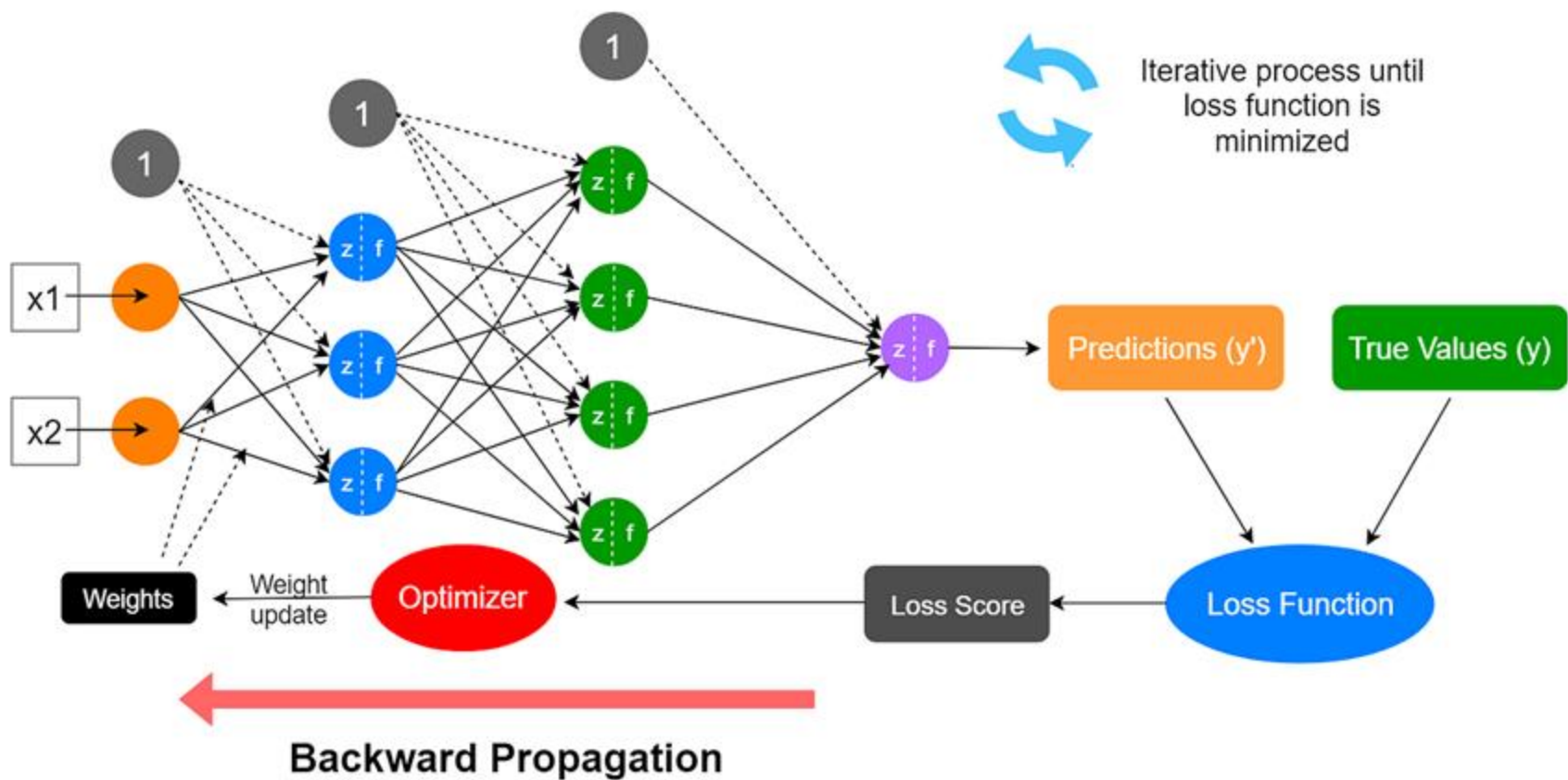


- We train the machine with thousands of samples so it can find **pattern**.
- The pattern can be modeled as a certain **function**
- This function will then used to predict the label of new images
- How can we create a good function to suit our purpose?



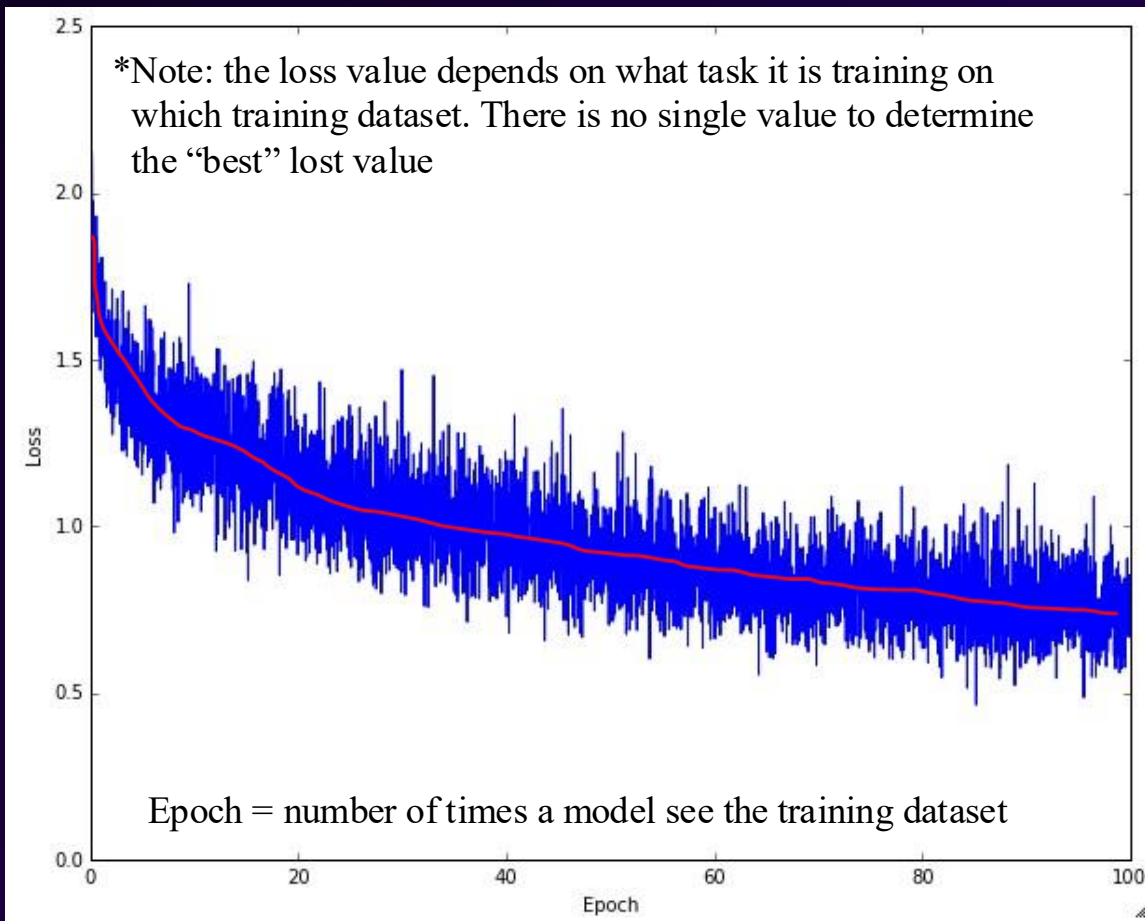


- Choose the best **a (weight)** and **b (bias)** to fit the line to the points. This is called **optimization**.
- We can apply the similar idea for finding patterns in images.
- How to optimize the parameters (weights + biases)? **Backpropagation!**





This is an example of a good (enough) loss function



# Overfitting & Underfitting

- Imagine you train the model with 1 image of Pomeranian on 1000 epochs for label “dog”.
- This is **overfitting**.
- Next time when you give the model an image of Husky, the model won't classify it as “dog”

Training



“dog”

Testing



Not a “dog”

# Overfitting & Underfitting

- Imagine you train the model with 1000 images of random animals on 1 epoch for label “animal”.
- This is **underfitting**.
- Next time when you give the model an image of an animal-looking monster, the model will classify as “animal”

Training



“animal”

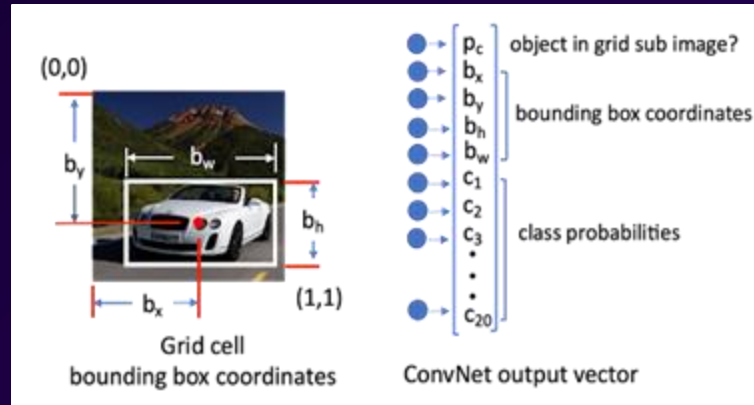
Testing



“animal”

# What about YOLOv8?

- The prediction (output) from YOLOv8 consists of **Objectness Score**, **Bounding Box Coordinates**, and each **Class Score**
- Then, it follows the training step: computing the loss against ground truth labels → adjust the weights and biases to improve accuracy
- This process repeats for multiple epochs until the prediction is sufficiently accurate.
- After training, **post-processing** is applied to the final predictions to filter overlapping boxes and return the final clean detection results

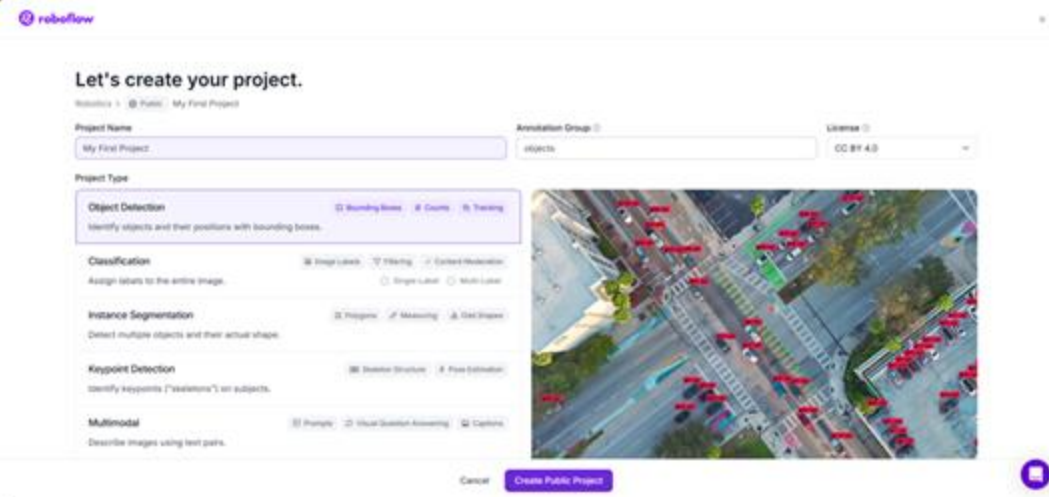


# Custom Your Own Dataset

- Closed-Set for specific task!
- Create your Own Label for objects

# Custom Your Own Dataset

- Platform – Roboflow
- Procedure - Startup



The image shows the Roboflow web interface for creating a new project. The header includes the Roboflow logo and a navigation bar with 'Resources', 'Public', and 'My First Project'. The main heading is 'Let's create your project.' Below this, there are three input fields: 'Project Name' (containing 'My First Project'), 'Annotation Group' (containing 'objects'), and 'License' (set to 'CC BY 4.0'). The 'Project Type' section is expanded, showing 'Object Detection' as the selected option. Other options include 'Classification', 'Instance Segmentation', 'Keypoint Detection', and 'Multimodal'. Each option has a brief description and a set of radio buttons for further configuration. For example, 'Object Detection' has options for 'Bounding Boxes', 'Coco', and 'Tracking'. The 'Object Detection' section also includes a preview image of a street scene with red bounding boxes around cars. At the bottom right, there is a 'Create Public Project' button and a 'Cancel' link.

Let's create your project.

Resources > Public > My First Project

Project Name: My First Project

Annotation Group: objects

License: CC BY 4.0

Project Type

**Object Detection** ☒ Bounding Boxes ☐ Coco ☐ Tracking  
Identify objects and their positions with bounding boxes.

**Classification** ☒ Image Labels ☐ Filtering ☐ Content Moderation  
Assign labels to the entire image.

**Instance Segmentation** ☒ Polygon ☐ Masking ☐ Mask Shapes  
Detect multiple objects and their actual shapes.

**Keypoint Detection** ☒ Skeleton Structure ☐ Pose Estimation  
Identify keypoints ("landmarks") on subjects.

**Multimodal** ☒ Prompt ☐ Visual Question Answering ☐ Caption  
Describe images using text pairs.

Cancel [Create Public Project](#)



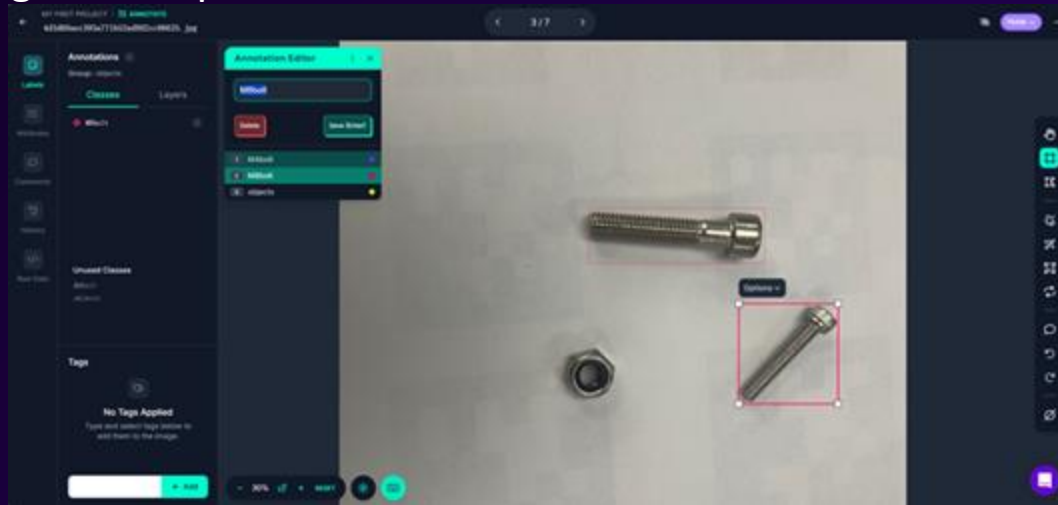
# Custom Your Own Dataset

- Procedure – Label
- Single / Multi objects



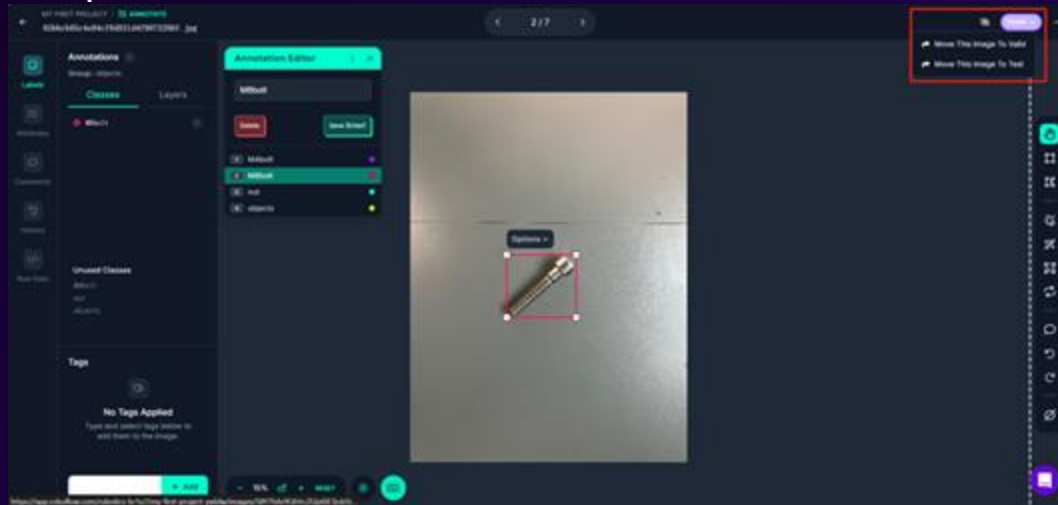
# Custom Your Own Dataset

- Procedure – Label
- Bounding Box Shape



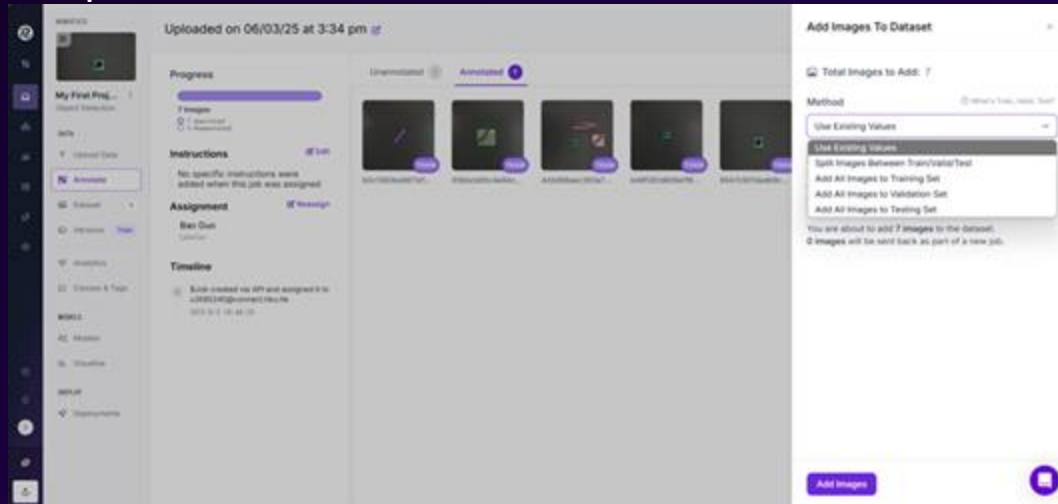
# Custom Your Own Dataset

- Procedure – Label
- Data for Purpose



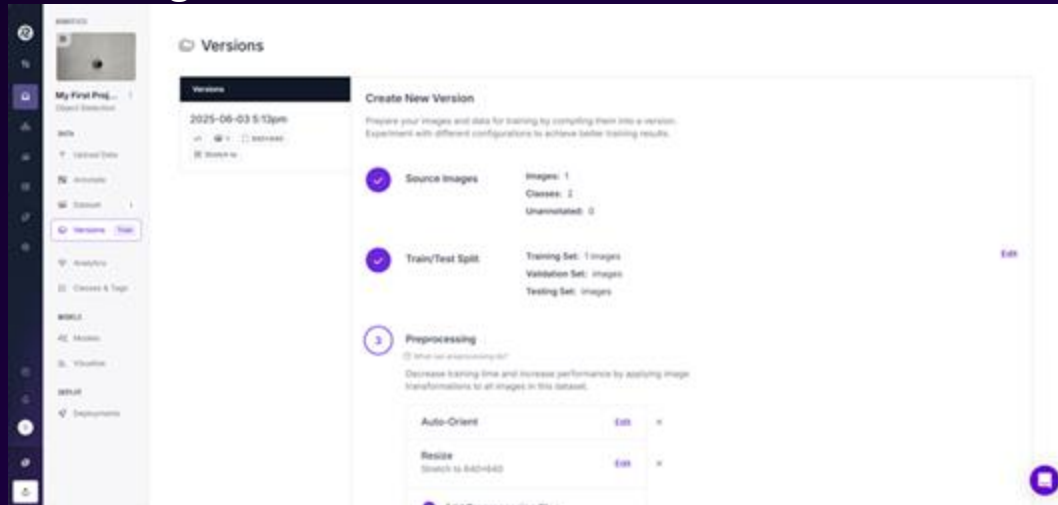
# Custom Your Own Dataset

- Procedure – Add to Dataset
- Data for Purpose



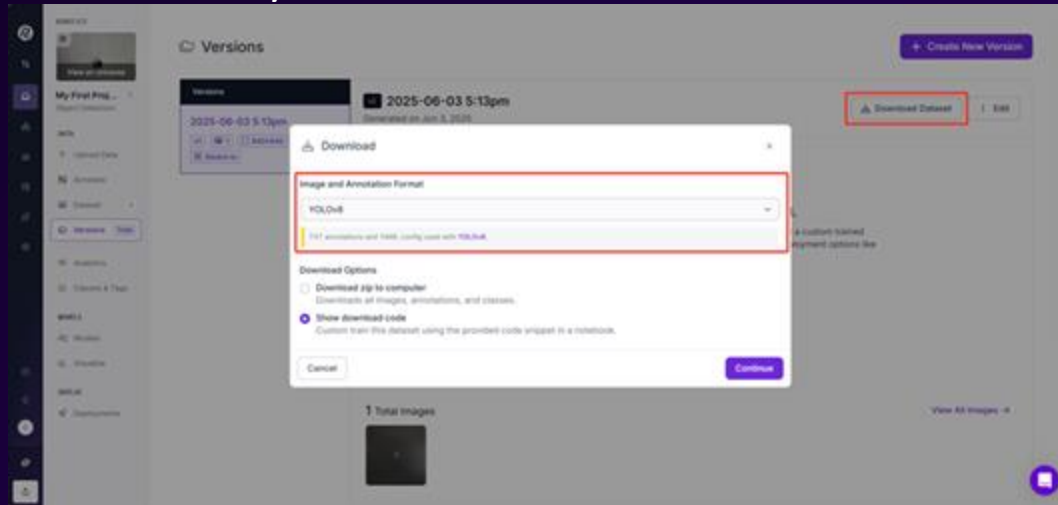
# Custom Your Own Dataset

- Procedure – Processing
- Preprocess & Augmentation



# Custom Your Own Dataset

- Almost Done!
- Select and download your dataset





# Custom Your Own Dataset

- Almost Done!
- Folder Structure

```
May not be exactly the same!

root\
|
|-- data.yaml
|-- README.dataset.txt
|-- README.roboflow.txt
|-- train\
|   |-- images\
|   |   |-- xxx.jpeg (.png, .jpg)
|   |   |-- ...
|   |   |-- xxx.jpeg
|   |-- labels\
|   |   |-- xxx.jpeg (.png, .jpg)
|   |   |-- ...
|   |   |-- xxx.jpeg
|-- val\
|   |-- ...
|-- test\
|   |-- ...
```

# Custom Your Own Dataset

- Reference: [Notion Page](#)

Thanks!!

Any questions?

