Yolo Guide

Last Update: June 2025

Table of Contents

[1 Prerequisites 3](#_Toc199867468)

[1.1 Python 3](#_Toc199867469)

[1.1.1 Check your python version 3](#_Toc199867470)

[1.2 Virtual Environment 3](#_Toc199867471)

[1.2.1 Windows 3](#_Toc199867472)

[1.2.2 Linux 4](#_Toc199867473)

[1.3 Packages and software requirements 4](#_Toc199867474)

[1.3.1 Windows 4](#_Toc199867475)

[1.3.2 Linux 5](#_Toc199867476)

[1.3.3 Cuda Toolkit 7](#_Toc199867477)

[2 Training Data 8](#_Toc199867478)

[2.1 Image Collection 8](#_Toc199867479)

[2.2 RoboFlow 9](#_Toc199867480)

[2.2.1 Create an account 9](#_Toc199867481)

[2.2.2 Create a project 9](#_Toc199867482)

[2.2.3 Add images to the project 10](#_Toc199867483)

[2.2.4 Annotate your images 10](#_Toc199867484)

[2.2.5 Review Annotations 15](#_Toc199867485)

[2.2.6 18](#_Toc199867486)

[2.2.7 Adding images to the dataset 19](#_Toc199867487)

[2.2.8 Create the final training set 20](#_Toc199867488)

[2.2.9 23](#_Toc199867489)

[2.2.10 Download the dataset 24](#_Toc199867490)

[3 Training a Model 25](#_Toc199867491)

[3.1.1 Create a model for python 25](#_Toc199867492)

[3.1.2 Convert Model to work with C++ 26](#_Toc199867493)

[4 Model Inference 27](#_Toc199867494)

[4.1 Python 27](#_Toc199867495)

[4.2 C++ 27](#_Toc199867496)

# Prerequisites

## Python

Install Python 3.8+ *(I used python 3.10 (windows) and 3.11 (raspberry py))*

[Install python here](https://www.python.org/downloads/)

### Check your python version

In the command prompt, run python --version to check which version you have installed.

## Virtual Environment

Navigate to a folder where you will run all yolo code and create a virtual environment. The virtual environment is a safe place for installing packages and running yolo.

If you skip this step, you may break your system packages and have issues later on.

### Windows

dir – command to print the contents of the current directory

cd – command to change to other directories

#### Create a virtual environment

Make sure you are in a folder you can locate later, the virtual environment will be created locally

python -m venv environment\_name

NOTE: Add “--system-site-packages” option to give venv access to globals

#### Open the virtual environment

First, navigate to the folder where the venv was created

.\*environment\_name*\Scripts\activate

#### Close the virtual environment

deactivate

### Linux

ls – command to print the contents of the current directory

cd – command to change to other directories

#### Create a virtual environment

Make sure you are in a folder you can locate later, the virtual environment will be created locally

python -m venv environment\_name

NOTE: Add “--system-site-packages” option to give venv access to globals

#### Open the virtual environment

First, navigate to the folder where the venv was created

source *environment\_name*/bin/activate

#### Close the virtual environment

deactivate

## Packages and software requirements

### Windows

#### Pip

Install pip if you do not already have it…

Update it if you do have it:

python -m pip install --upgrade pip

#### Ultralytics

pip install ultralytics

#### PyTorch

Use this website <https://pytorch.org/get-started/locally/> to get the correct command.

pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu128

I selected version 12.8 because my cuda drivers require it later on…

|  |
| --- |
| A screenshot of a computer  AI-generated content may be incorrect. |

### Linux

#### Pip

Install pip if you do not already have it…

Update it if you do have it:

python -m pip install --upgrade pip

#### Ultralytics

pip install ultralytics

#### PyTorch

Use this website <https://pytorch.org/get-started/locally/> to get the correct command.

pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu128

I selected version 12.8 because my cuda drivers require it later on…

|  |
| --- |
| A screenshot of a computer  AI-generated content may be incorrect. |

### Cuda Toolkit

Install the Cuda Toolkit here <https://developer.nvidia.com/cuda-downloads>

NOTE: you may require an older version if you have an older gpu

# Training Data

The training data consists of images with bounding rectangles around the objects in the image you wish to identify. No knowledge of computer vision required.

## Image Collection

I used my robot’s camera to collect around 100 images of the Evac Zone from a few different angles. The images contain a mix of one or more of the objects.

|  |  |  |
| --- | --- | --- |
| Balls on a white surface  AI-generated content may be incorrect. | A group of balls on a white table  AI-generated content may be incorrect. | A ball on a white surface  AI-generated content may be incorrect. |
| A group of marbles on a white surface  AI-generated content may be incorrect. | A ball on a green triangle  AI-generated content may be incorrect. | A group of colorful objects on a white surface  AI-generated content may be incorrect. |

The images can contain full or partial views of the objects.

Hint: this can be done quickly with a program that saves an image when you press space bar

## RoboFlow

Per Ultralytics suggestion, I used roboflow to create the training data.

*I previously wrote a program to create the training data manually, but RCJ does not require development anymore so I will be showcasing a much simpler method.*

### Create an account

Create an account [here on roboflow](https://roboflow.com/)

### Create a project

|  |  |
| --- | --- |
|  | Create a new project in the Project tab |
|  | Select **Object Detection** as the project type.  Name the project and give a name to the group of items you can trying to detect (i.e. *victims*) |

### Add images to the project

|  |  |
| --- | --- |
|  | Upload the folder with all the images captured on from your robot |

### Annotate your images

Annotating your images is the process of identifying the objects in your uploaded images. All you need to do is draw bounding boxes around your images and specify the class if belongs to.

I will share some helpful notes for annotating.

#### How to get started

|  |  |
| --- | --- |
|  | Navigate to the **annotate** page and select **annotate images** in the top right |
|  | When prompted, select to annotate the images manually. This is the easiest way to annotate the first time |

#### Annotation Interface

|  |  |
| --- | --- |
|  | To annotate, simply click and drag to make boxes around the object you wish to detect later on. |
|  | Make sure each bounding box you make is associated with the correct class label.  You can create any class name you want with the ***annotation editor***. |
|  | Once your classes are created, you can use the keyboard number and the enter key to quickly classify the objects in your images.  ***OR*** *you can just click the correct class with your mouse* |
|  | The ***drag tool*** on the left side can be used to select and modify any of the bounding rectangles you have made. |
|  | Select the ***bounding box tool*** again to begin creating bounding boxes. |
|  | Use the arrows at the top of the screen to move between images. |

#### Annotation Tips

|  |  |
| --- | --- |
|  | Use *tight* bounding boxes |
|  | Do not cut off any of the object!! |
|  | Annotate objects even if they are only partially visible |
|  | Annotate objects even if they are overlapping |

### Review Annotations

When you are done annotating the images, you must submit them for “review”.

Note: for the demo, I did not annotate the full set of images, but **it is highly recommended that you annotate all of your images**

|  |  |
| --- | --- |
|  | You can easily take a look at your annotated images.  **NOTICE:** all images say “Train”… this will be modified later automatically when the training set is split |

#### Submit for review

This process is designed for groups of people to work on a project together. You will need to review your own annotations and approve them yourself.

|  |
| --- |
|  |

|  |
| --- |
|  |

#### Review and apporve one at a time

Select one of the images in the “**To Do**” tab and you can begin reviewing the images one at a time.

You can make any modifications you want to fix the annotations.

|  |  |
| --- | --- |
|  | Select any image to start the review |
|  | Modify the annotations as needed.  Then approve the image when you are satisfied. |

#### Review and apporve all at once

|  |  |
| --- | --- |
|  | You can approve all annotated images at once and then add them to the dataset. |

### 

### Adding images to the dataset

|  |  |
| --- | --- |
|  | Check your approved images, if you are satisfied you can add the approved images to a dataset. |

|  |  |
| --- | --- |
|  | ***IMPORTANT:*** select the method that will automatically split your dataset into the 3 catagories “train” “valid” and “test”  Train – training images for the model  Valid – images used during training to validate the progress of the model and provide feedback  Test- images to test the model’s performance after the training is complete |

### Create the final training set

Pay close attention to the options shown below!!

**You can choose to modify these optional settings as you see fit for your training set.**

*You can redo this at any time with new options selected.*

|  |  |
| --- | --- |
|  | Navigate to the “**Versions**” tab. |

#### Preprocessing

Modifications to the image before being passed to the mode. By default, roboflow does nothing to the image and simply passes in a 640, 640 image.

Modify the images in the pre-processing steps and make sure you pass in the same images for inference that the model was trained on.

|  |  |
| --- | --- |
|  | Resize the images to 320x240 for training.  This is to increase model speed.  I use 320x240 on my robot as well to increase program speed. |

#### Augmentation

|  |  |
| --- | --- |
|  | The current training images can be augmented and modified to create new training images for different environmental conditions. |
|  | Adds images to the training dataset with augmented lighting conditions.  You can select how much the dataset will be augmented. |
|  | You can add more augmented images if you would like |

#### Finalize the dataset

|  |  |
| --- | --- |
|  | Finalize and create the dataset.  Select the number of variations you would like to create for each of your augmentation steps.  **NOTE:** if you do not select any augmentations, you will not have this option. |

### 

### Download the dataset

Once you are done, a dataset will be available for download.

|  |
| --- |
|  |

|  |  |
| --- | --- |
|  | Select the model YOLOv8 (or whichever model you are using).  Download the data locally, do not train a model yet… |

Download and ***extract the training data*** to any folder you want

# Training a Model

To train the model I ran the program on a computer different from my robot.

I am using visual studio code on a computer with a Nvidia GPU. *Training times will be significantly slowing if you train using a cpu*

### Create a model for python

First, create a model for python use (.pt pytorch model) and then if you want, you can convert it later for use with a c++ file

#### Virtual Environment

Make sure you have the correct virtual environment selected.

|  |  |
| --- | --- |
|  | Select the venv in the bottom right corner of VS code |

#### Data path

To train a model with your own custom dataset, you need to specify the correct path to the data.

If you open the folder that you downloaded, you should see a **data.yaml** file. This is the file needed to train a model.

|  |
| --- |
|  |

#### Code (GPU TRAINING)

|  |  |
| --- | --- |
| from ultralytics import YOLO  if \_\_name\_\_ == "\_\_main\_\_":      data\_path = “path to training data as a string”      # using a model from yolov8      model = YOLO('yolov8n.yaml')  # custom training      results = model.train(data = data\_path,                            imgsz = 320,                            device = 0,                            pretrained=False)    # save the pytorch model      model.save("model100.pt") | Import the necessary libraries  *THIS NEEDS TO BE INSIDE* a **python main** or it wont work  **PATH IS TO THE data.yaml file**  Create a model based on Yolov8  Train the model with your own custom data using a path to your data from roboflow  imgsz = 320 because images from my robot will be 320x240  device = 0 means GPU training (you can use device = “cpu”as well)  pretrained = False trains the model from scratch  Save the pytorch mode. You can name it whatever you want. |

### Convert Model to work with C++

.pt models work with python, not c++

Convert the model to an **onnx** file to work in **c++**

|  |  |
| --- | --- |
| from ultralytics import YOLO  if \_\_name\_\_ == "\_\_main\_\_":      model = YOLO("model100.pt")      model.export(format = "tflite", imgsz = 320) |  |

# Model Inference

## Python

|  |
| --- |
|  |

## C++

Add this to the geany **Build > Set Build Commands**

Add this to the end of **C++ Commands (Build)**

*-l:libopencv\_dnn.so*

A screenshot of a phone

AI-generated content may be incorrect.

|  |
| --- |
|  |