How to setup Python environment and execute yolov8 object detection machine learning model for license plate recognition

(The latest version of the project 2023-09-04 |done by Ahmed balfqiah and Abdullah Alshateri)

Chapter 1: Setting Up a Python Environment in PyCharm

1. Installing Python:

Go to the official Python website's download page:

https://www.python.org/downloads/Download

Open the downloaded .exe file to start the installation.

Click "Install Now" and wait for the installation to complete.

2. Installing PyCharm:

Go to the official JetBrains website's PyCharm download page:
https://www.jetbrains.com/pycharm/download/?section=windows

Choose the version you want to download the Community version

*** *you will find the community version after scrolling down on the website.*

Download the installer for your operating system (Windows's \macOS\Linux's)

Run the installer and follow the on-screen instructions to install PyCharm.

After installation, you can open PyCharm and start a new project. When prompted, choose the installed Python interpreter as the project interpreter. This will allow PyCharm to recognize and work with the Python version you've installed.

3. Installing the pip packages:

Open command prompt and enter the following commands one **after the other**:

pip install opencv-python

pip install requests

pip install numpy

pip install matplotlib

pip install yolov5

4 . open the project

- Locate the Zip File on Your Desktop
- Extract the Zip File
- Open PyCharm
    - *If you already have a project open, close it by going to File > Close Project.*
- Open the Extracted Project in PyCharm
    - On the welcome screen, you'll see a pane on the left side labeled "Projects".
    - At the top of this pane, click the file icon (which looks like a folder).
    - A file explorer window will pop up.
    - Navigate to your desktop and *select the folder you've just extracted*.
    - Click OK or Open.

- Your extracted project will be loaded into PyCharm's workspace.
- The project step is done

******* Handling Missing Packages in PyCharm:***

- Look at the First Lines of the code: you'll have your imports. Any package that isn't recognized will be underlined in red.

- Hover Over the Red Underline: A tooltip should pop up indicating "Package not found".

- Install Directly:
    - Right-click on the red-underlined package name.
    - From the dropdown, select Install package <package-name>.

- Allow Installation: PyCharm will handle the installation. Once finished, the red underline will disappear.

**** Do this for each red-underlined package on the import line.*

Chapter 2: License Plate Detection and Text Extraction

1. *Overview*

This program identifies license plates in vehicle images and extracts their alphanumeric text and the state of issuance.

2. *Dependencies*

os: For path manipulations when accessing model weights.
cv2: OpenCV, to process and manipulate image data.
requests: To retrieve image data from provided URLs.
numpy: For numerical operations on image data.
io: To handle byte streams, useful for processing image data retrieved from URLs.
matplotlib: To display the processed images.
ultralytics: Contains the YOLO model for object detection.

3. *Configuration*

a. Model Path

```
# Set the path to the model weights
model_path = os.path.join('.', 'runs', 'detect', 'train23', 'weights', 'last.pt')
```

*** The relative path to the YOLO model weights | train(xx) is changing depending on the model*

b. Class Definitions

```
# Define the specific classes of interest
specific_classes = ['0', '1', '2', '3', '4', '5', '6', '7', '8',
                    '9', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',
                    'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q',
                    'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']
```

*** specific_classes: A list of alphanumeric characters that could appear on license plates.*

*** state_classes: A dictionary to map abbreviated state class names (from model predictions) to their full names.*

```
# Mapping state classes to their full names
state_classes = {
    'exp': 'Export',
    'new_DUBAI': 'Dubai',
    'new_RAK': 'Ras Al-Kaimah',
    'new_abudabi': 'Abu Dhabi',
    'new_ajman': 'Ajman',
    'new_am': 'Umm Al Quwain',
    'new_fujairah': 'Fujairah',
    'old_DUBAI': 'Dubai',
    'old_RAK': 'Ras Al-Kaimah',
    'old_abudabi': 'Abu Dhabi',
    'old_ajman': 'Ajman',
    'old_am': 'Umm Al-Quwain',
    'old_fujira': 'Fujairah',
    'old_sharka': 'Sharjah'
}
```

c. Model and Threshold Initialization

```
# Load the YOLO model with the specified model path
model = YOLO(model_path)

# Define the detection threshold
threshold = 0.4
```

*** Loading the YOLO model and setting a detection confidence threshold.*

4. *Core Functions*

a. detect_text(image)

i. Objective

This function's primary goal is to detect and identify characters on a given license plate image.

ii. How it works

1. **Image Shape Extraction**
   Grabs the dimensions (Height, Width) of the input image. This might be used in scenarios where coordinates need to be processed with respect to the image size.

2. **Model Detection**
   The model processes the image to find bounding boxes – rectangular areas where the characters might be located.

3. **Sorting Detected Characters**
   The detected results are sorted based on the x-coordinate. This ensures that the characters are processed from left to right, preserving their order on the license plate.

4. **Extracting Characters**
   For each bounding box:
   - The class of the detected object and its confidence score is extracted.
   - If the detected object is one of the predefined alphanumeric characters and the confidence score is above the threshold, it's added to the output list.

5. **Identifying the State**
   A similar process is followed to identify the state, but here it checks against a list of states instead of alphanumeric characters. If a state is identified with confidence above the threshold, the state's full name is extracted from the state_classes dictionary.

6. **Error Correction**
   There is a specific error correction step. If both 'D' and 'Q' are detected together, it considers it an error and removes 'Q' from the output list.

iii. **Parameters**

image: This is the input image, ideally a close-up of a license plate. It should be provided in a format that the model can process, typically a multi-dimensional array.

iv. **Returns**

1. A list of detected alphanumeric characters from the license plate in the order they appeared.
2. he identified state name, or a message saying it couldn't be detected.

b. main ()

i. **Objective**

This function serves as the primary driver of the script. It processes online images, detects license plates, highlights them, and prints the extracted text and state.

ii. **How it works**

1. **Fetching Images**
   - The function iterates through a list of image URLs.
   - For each URL, it attempts to download the image using the requests library. If successful, it processes the image to be suitable for the model.

2. **License Plate Detection**
   - The model processes the entire vehicle image to detect objects.
   - If a license plate ('plate') is identified with confidence above the threshold, its coordinates are extracted.

3. **Highlighting Detected Plates**

   Using OpenCV (cv2), the script draws a green rectangle around the detected license plate on the original image**.**

4. **Text Extraction**
   - The detected license plate area (bounded by the green rectangle) is cropped from the original image.
   - This cropped image (the license plate) is then passed to the detect_text function. This returns the detected alphanumeric characters and the identified state.

5. **Displaying Results**
   - The extracted information (alphanumeric text and state) is printed to the console.
   - The original image, with the highlighted license plate, is displayed using matplotlib.

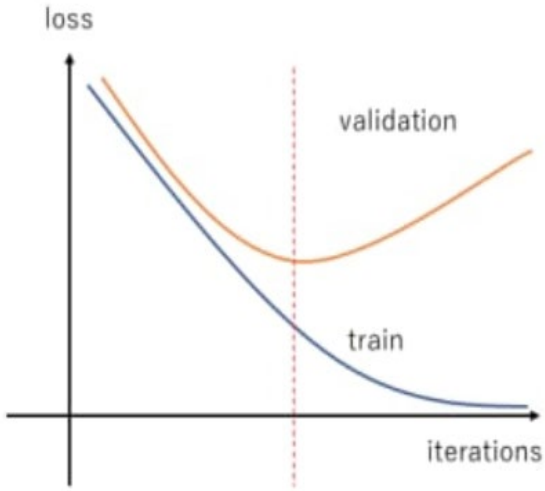
5. *Evaluating Our Machine Learning Model and Demonstrating Results*

Overfitting:

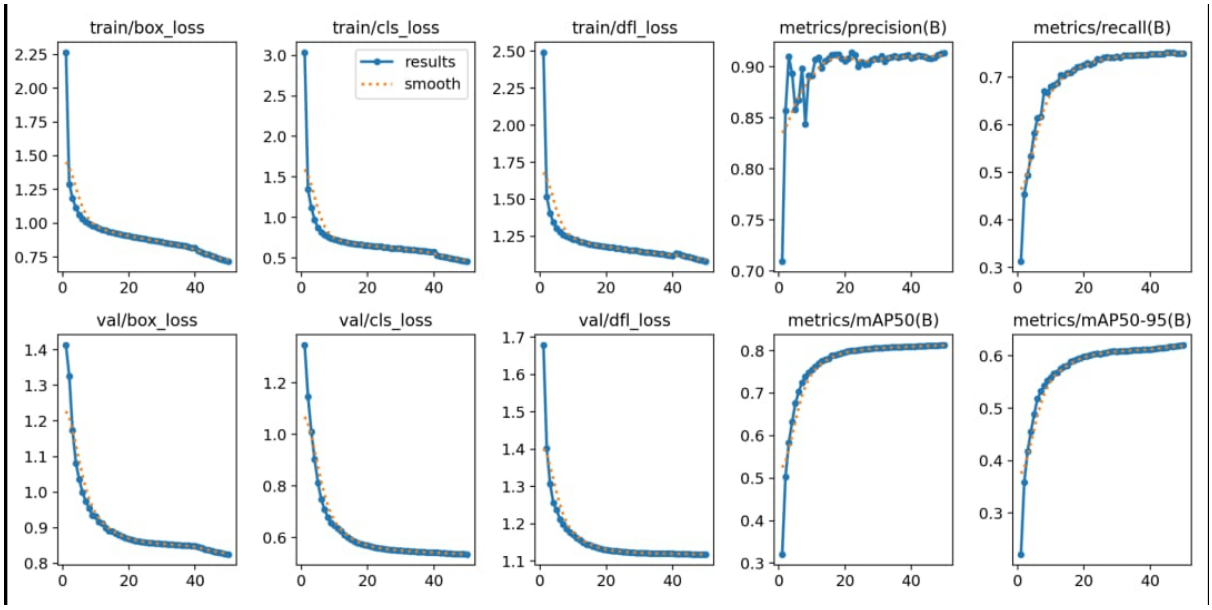
Overfitting happens when a model learns the training data too well, including its noise and outliers. As a result, it performs poorly on new, unseen data. Imagine trying to memorize answers for a test instead of understanding the subject. While you might do well on that specific test, you'll likely fail at solving new problems on the same subject. In machine learning, a model that overfits has essentially done the same thing – it's "memorized" the training data instead of "understanding" the underlying patterns.

To avoid overfitting:

- Use more data: If possible, increase the amount of training data.
- Regularization: Techniques like L1 and L2 regularization can constrain the model's complexity.
- Cross-validation: Use subsets of the training data to validate the model's performance.
- Simplifying the model: Sometimes, using a less complex model or reducing the number of features can help.
- Early stopping: In iterative algorithms, stop training before the model starts to overfit.
- Dropout: In neural networks, dropout is a technique where randomly selected neurons are ignored during training.



*** Identifying overfitting is crucial for developing robust models that perform well not just on their training data but on unseen data as well.*



Simplified Explanation of Terms:

Box_loss:

This represents the error related to determining the position and size of bounding boxes around objects in the image.

Cls_loss:

This pertains to the error in classifying the objects within those bounding boxes, i.e., determining what type of object is contained within the box.

Dfl_loss (Distribution focal loss):

This is a more advanced loss function that is specifically designed to address class imbalance. It assigns more weight to the less frequent classes, ensuring the model doesn't get dominated by more prevalent classes.

*** These terms are often used in the context of object detection models to evaluate and improve their performance.*

Datasets:

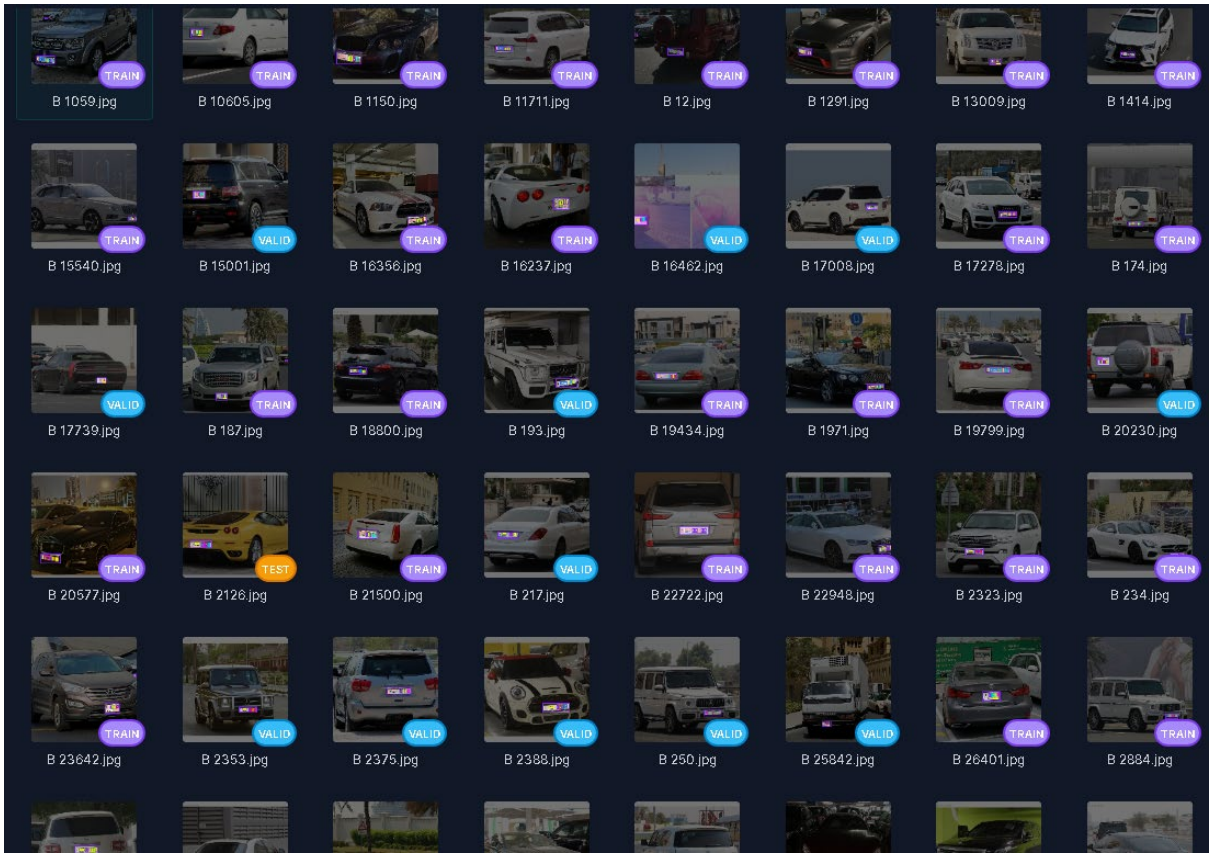
A dataset is a collection of data points, often used to train, test, and validate machine learning models. It usually comprises of features (inputs) and labels (outputs or what you're trying to predict).

- **Train Set**
  The training set is a subset of your dataset used to train your machine learning model. By "train," we mean the process by which the model learns patterns from the data. Most of the dataset (often 60-80%) is typically allocated to the training set.

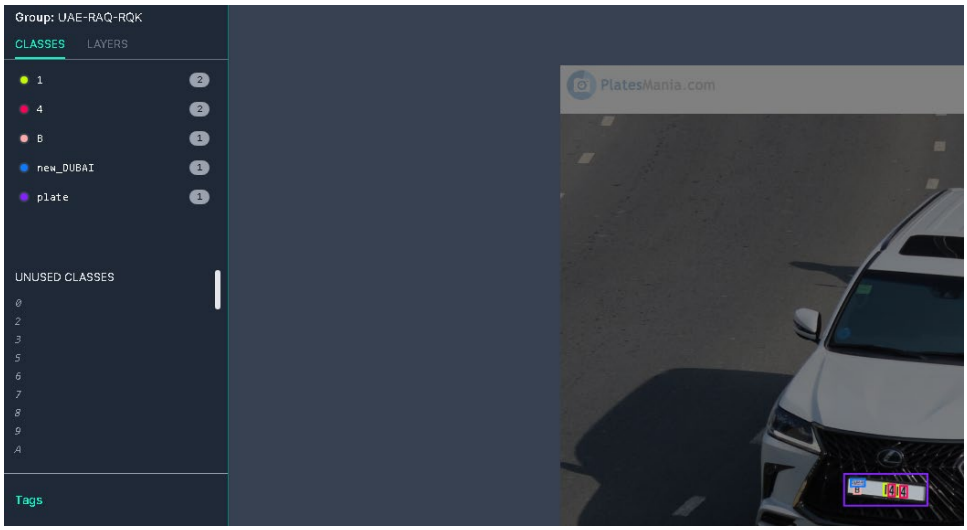
- **Test Set**
  After training a model, you need to know how well it's likely to perform on new, unseen data. This is where the test set comes in. It's a subset of your dataset that you set aside and don't use during training. Instead, after training is complete, you run the model on the test set to see how well it performs. This gives an indication of how the model might perform in the real world.

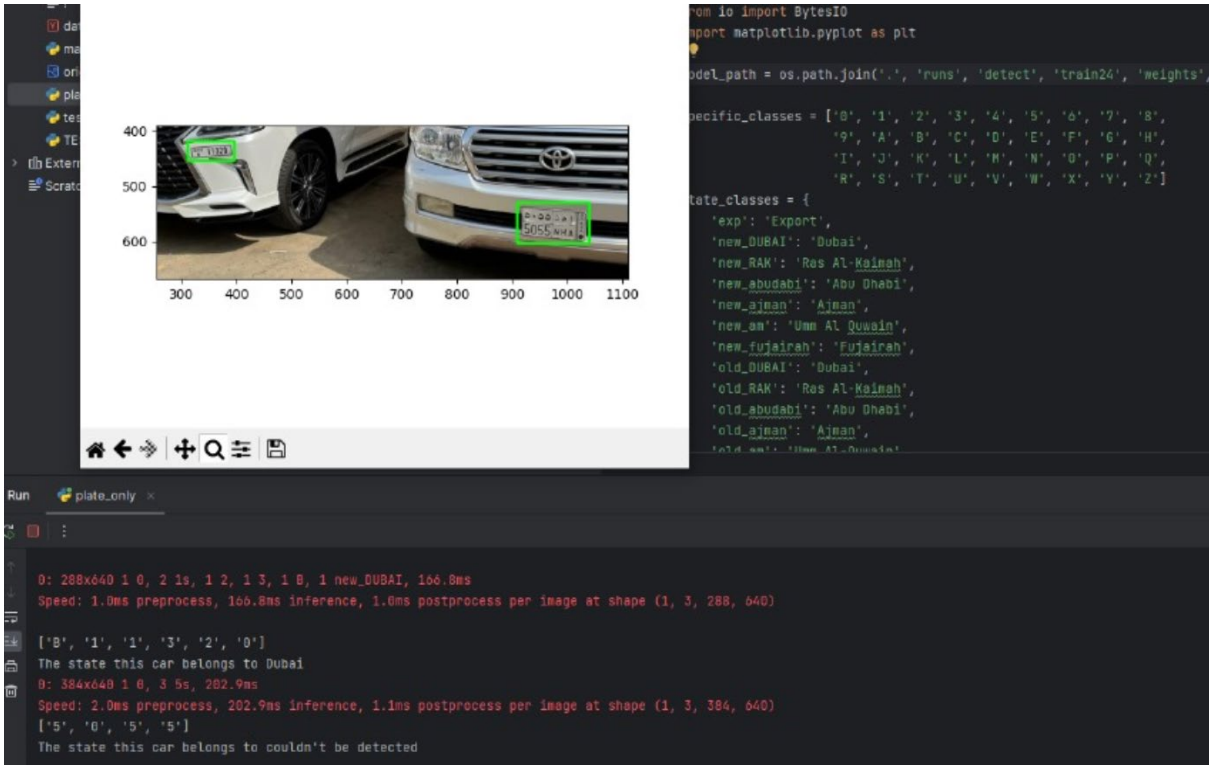
- **Validation Set**

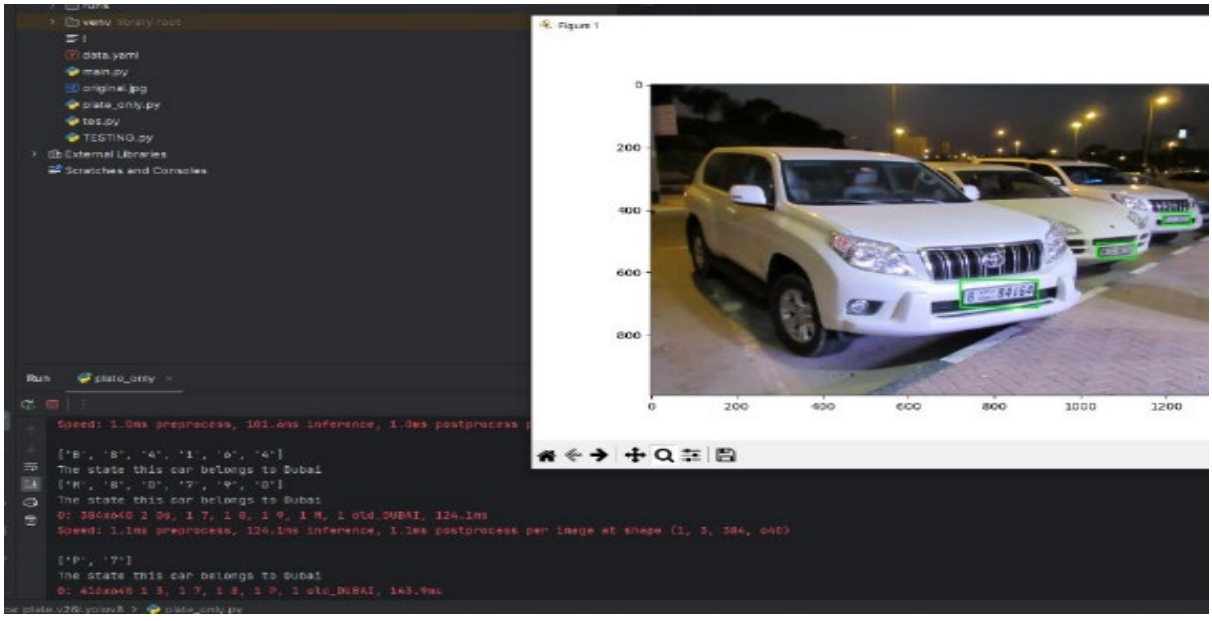
  Sometimes, during training, you might want to tweak your model's parameters to improve performance. But if you keep tweaking your model to perform well on the test set, you run the risk of overfitting to that test set. The solution? Use a validation set. It's another subset of your dataset that you can use to evaluate performance during training, without touching the test set. Once you're satisfied with the model's performance on the validation set, you can do the final evaluation on the test set.



Results:


Group: UAE-RAQ-RQK
CLASSES
LAYERS
1
2
4
2
B
1
new_DUBAI
1
plate
1
UNUSED CLASSES
Tags
PlatesMania.com


Run
plate_only
0: 288x640 1 0, 2 1s, 1 2, 1 3, 1 B, 1 new_DUBAI, 166.8ms
Speed: 1.0ms preprocess, 166.8ms inference, 1.0ms postprocess per image at shape (1, 3, 288, 640)
['B', '1', '1', '3', '2', '0']
The state this car belongs to Dubai
0: 384x640 1 0, 3 5s, 202.9ms
Speed: 2.0ms preprocess, 202.9ms inference, 1.1ms postprocess per image at shape (1, 3, 384, 640)
['5', '0', '5', '5']
The state this car belongs to couldn't be detected


data.yaml
main.py
original.jpg
plate_only.py
tes.py
TESTING.py
External Libraries
Scratches and Consoles
Figure 1
Run
plate_only
['B', '8', '4', '1', '6', '4']
The state this car belongs to Dubai
['M', '8', '0', '9', '9', '0']
The state this car belongs to Dubai
['P', '7']
The state this car belongs to Dubai