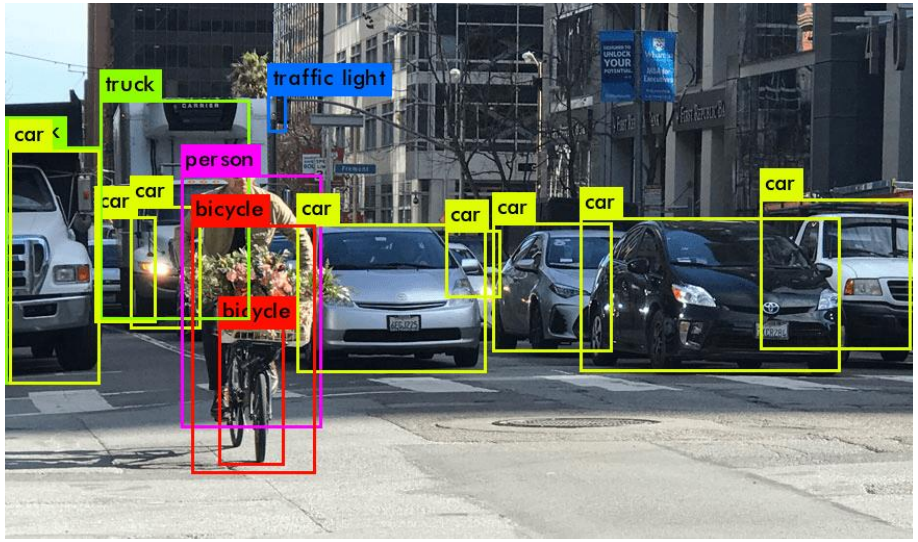
DEEP LEARING

MINI PROJECT

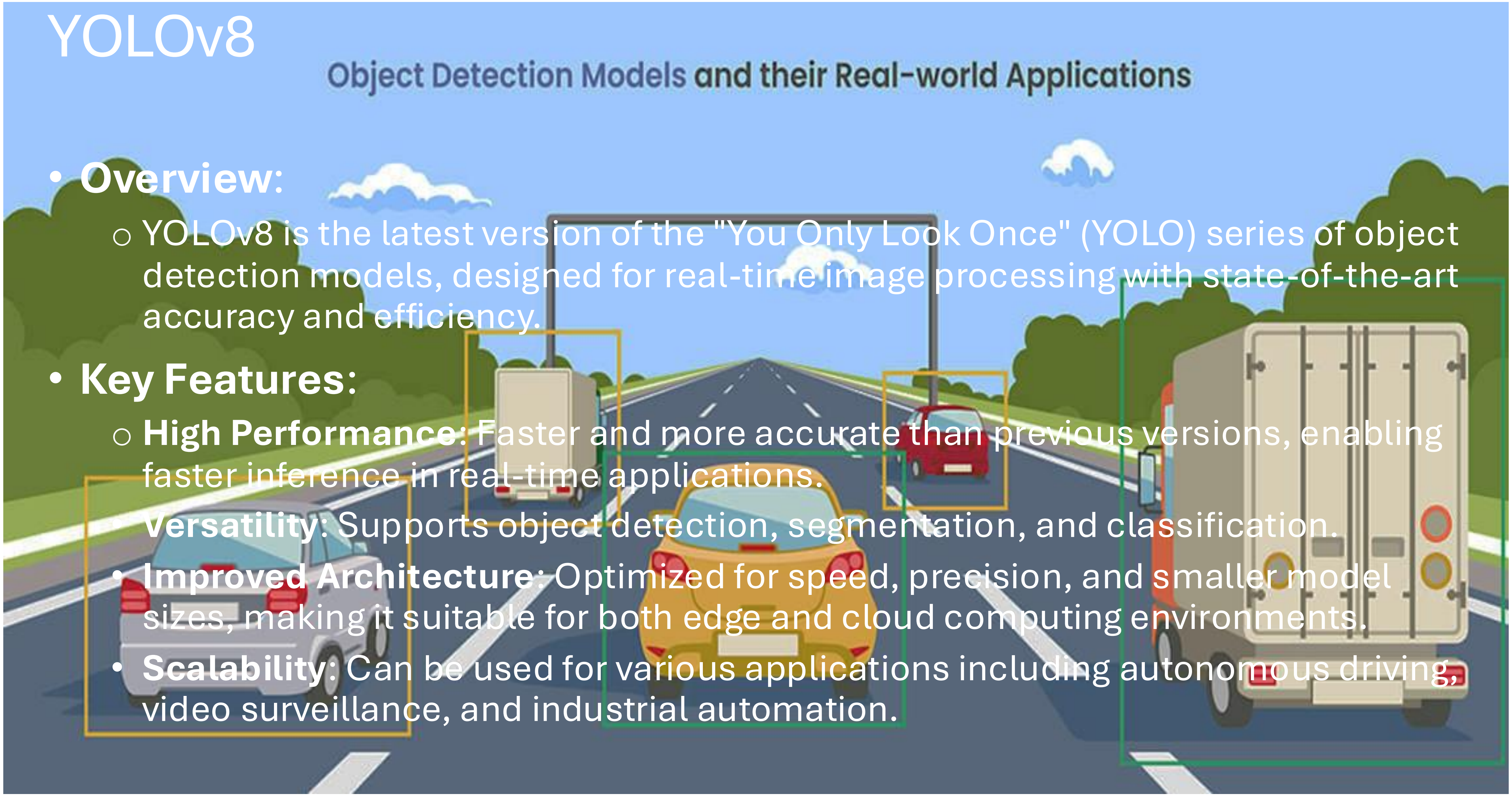
ID-CARD DETECTION

ID-CARD DETECTION
Using YOLOv8

Object Detection on a Custom Dataset



YOLOv8



- **Overview:**
  - YOLOv8 is the latest version of the "You Only Look Once" (YOLO) series of object detection models, designed for real-time image processing with state-of-the-art accuracy and efficiency.
- **Key Features:**
  - **High Performance:** Faster and more accurate than previous versions, enabling faster inference in real-time applications.
  - **Versatility**: Supports object detection, segmentation, and classification.
  - **Improved Architecture**: Optimized for speed, precision, and smaller model sizes, making it suitable for both edge and cloud computing environments.
  - **Scalability**: Can be used for various applications including autonomous driving, video surveillance, and industrial automation.

Applications & Advantages

Applications:

- Autonomous vehicles
- Security systems
- Robotics
- Augmented Reality (AR)

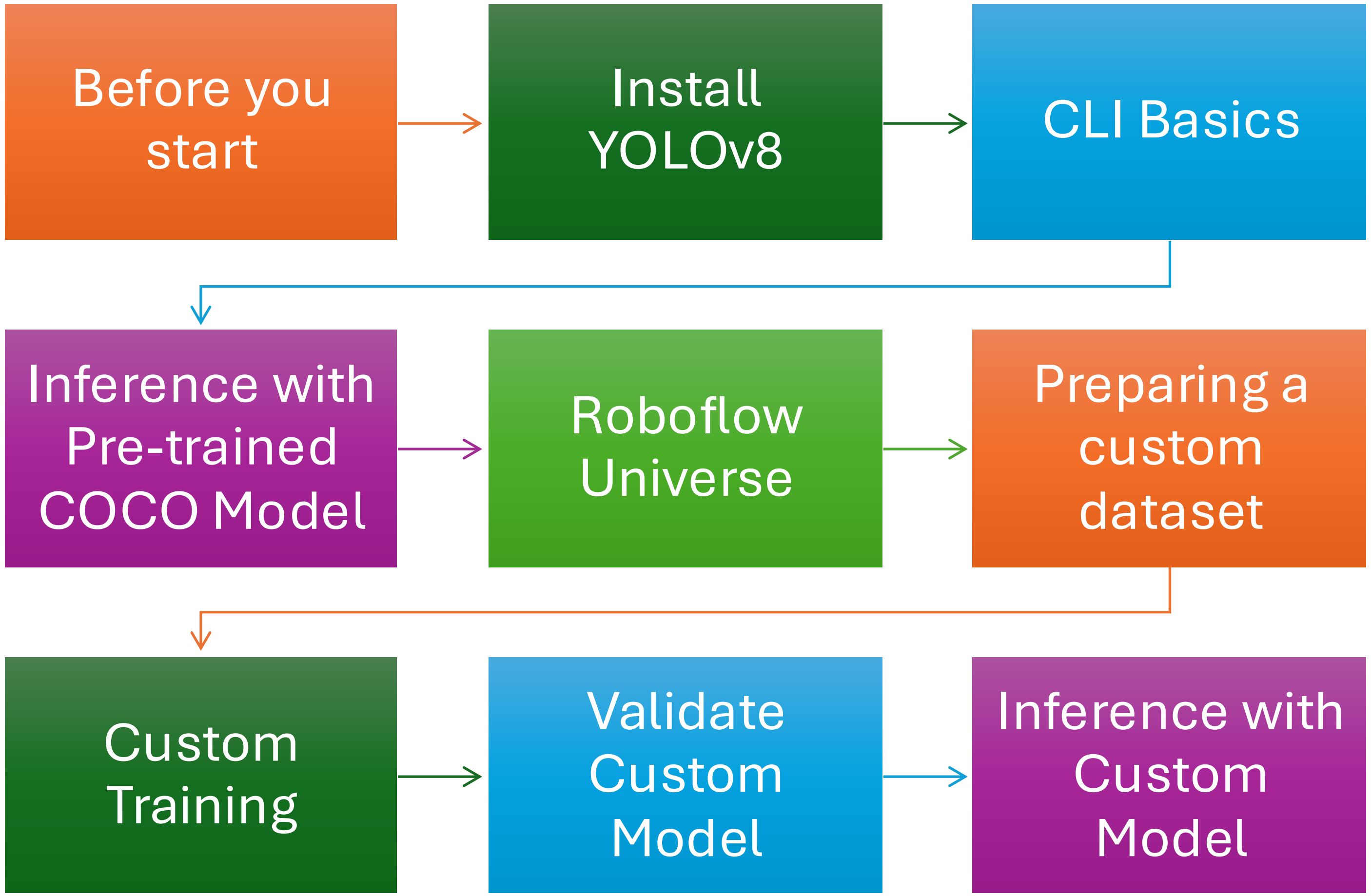
Advantages:

- Faster inference times
- Better generalization to unseen data
- Smaller model size for deployment in real-world applications

PROCESS

- Data Collection
- Data Preprocessing
- Annotation
- Model Selection
- Model Training
- Evaluation
- Model Deployment

Steps


Before you start
Install YOLOv8
CLI Basics
Inference with Pre-trained COCO Model
Roboflow Universe
Preparing a custom dataset
Custom Training
Validate Custom Model
Inference with Custom Model

Data Collection and Labeling using Roboflow

Data Collection:

- **Upload Images**: Collect and upload your image dataset to Roboflow. Supported file formats include .jpg, .png, .jpeg, etc.
- **Data Diversity**: Ensure that the dataset includes various angles, lighting conditions, and object appearances for better model generalization.

Automatic Data Augmentation:

- **Augmentation Options**: Roboflow allows automatic augmentation of your dataset, including transformations like rotations, flips, color adjustments, and zooming.
- **Increase Dataset Size**: Augmentation increases dataset size and variability without requiring additional manual data collection.

Labeling the Dataset & Formating

Labeling the Dataset:

Manual Labeling: Use the Roboflow interface to manually annotate objects in images by drawing bounding boxes and assigning class labels.

Auto-Labeling: If pre-labeled data is available, Roboflow can auto-label new images based on the existing dataset.

Labeling Tools: Roboflow provides a user-friendly tool to draw bounding boxes around objects and categorize them with predefined classes.

Labeling Formats:

YOLO Format: Bounding boxes are stored as .txt files, where each line contains the class ID and normalized coordinates of the bounding box.

Other Formats: You can also label in other formats like COCO, Pascal VOC, etc., depending on your model's requirements.

Exporting dataset:

```
!mkdir -p {HOME}/datasets
%cd {HOME}/datasets
!pip install roboflow
from roboflow import Roboflow
rf = Roboflow(api_key="ZtmYcD2C7omgXhd1ovm2")
project = rf.workspace("nithwin").project("idcard_2_0")
version = project.version(7)
dataset = version.download("yolov8")
```

Custom Training

```
%cd {HOME}

!yolo task=detect mode=train
model=yolov8s.pt
data={dataset.location}/data.yaml
epochs=7 imgsz=500 plots=True save=True
```

- **task=detect**: Specifies that the task is object detection.
- **mode=train**: Initiates the training process.
- **model=yolov8s.pt**: Uses the pre-trained YOLOv8 small model (yolov8s.pt) as a starting point for training.
- **data={dataset.location}/data.yaml**: Specifies the path to the dataset configuration file (data.yaml) that includes details about class names and dataset paths.
- **epochs=7**: Specifies 7 epochs for training (can be adjusted for more/less iterations).
- **imgsz=500**: Sets the image size to 500x500 pixels for input images.
- **plots=True**: Enables the generation of plots during training (e.g., loss curves).
- **save=True**: Saves the model checkpoints after training.

Validate Custom Model

```
%cd {HOME}

!yolo task=detect mode=val model={HOME}/runs/detect/train3/weights/best.pt data={dataset.location}/data.yaml
```

This command runs YOLO in validation mode (mode=val) to test a trained object detection model (best.pt).

It uses the dataset specified in the data.yaml file, which contains information about the dataset's images and classes.

The model is located in the {HOME}/runs/detect/train3/weights/ directory.

Deploy model on Roboflow

Inference with Custom Model

project.version(dataset.version).deploy(model_type="yolov8", model_path=f"{HOME}/runs/detect/train/")

The command deploys a YOLOv8 model from the specified directory for the dataset version you're working with.

- **`project.version(dataset.version)`**: This specifies the version of the dataset you are using for the deployment. The version is usually defined in your project to track different stages of the dataset.
- **`deploy(model_type="yolov8", model_path=f"{HOME}/runs/detect/train7")`**: This part is deploying the model with the specified parameters:
  - **`model_type="yolov8"`**: This indicates you're using the YOLOv8 model version.
  - **`model_path=f"{HOME}/runs/detect/train/"`**: This specifies the directory where the trained YOLOv8 model is stored, using the path `{HOME}/runs/detect/train/`.In short:


THANK YOU