CSE 424 Project Presentation

Obstacle Pattern Recognition From Vehicle Perspective

Section: 01

Group Number: 23

Group Members:

1. Md. Imamul Mursalin Sujoy (20301120)
2. Md. Asif Rahman (20301122)
3. Jannatul Ferdoush (20301291)
4. Utsha Sen Dhruba (20301338)

Goals

Object detection in a way which is more efficient and reliable for autonomous vehicle or mobile robots.

The detection of objects are -

- Pedestrians
- Cars
- Cyclists
- Animals
- Electric pole

Object detection procedures -

- Localization
- Classification

Literature Review

Paper: You Only Look Once : Unfield, Real-Time Object Detection

- The paper presents a unifield , real-time object detection system that achieves fast processing speeds and high generalization
- YOLO enables end-to-end training and achieves real-time speeds, processing up to 155 frames per second.
- It generalize top detection methods like DPM and R-CNN when tested on different domains, such as artwork.
- The model have difficulties localizing small objects and can produce localization errors.

Literature Review

Paper: Object Detection Using Convolutional Neural Networks

- To implement the object detection using CNN, TensorFlow Object detection API was used which is an open source framework for object detection models.
- Two state of the art models are compared for object detection
- SSD with MobileNetV1 has high speed detection but low accuracy
- Faster-RCNN with InceptionV2 has low speed but more accurate detection

Literature Review

Paper: Faster R-CNN: an Approach to Real-Time Object Detection

- This model merges RPN and Fast R-CNN thus it offers improved speed and accuracy
- Four classes, allocating 80% for training and 20% for testing
- The paper have showed good potential on identifying traffic signal on real time using cheap dashboard camera.
- The model does not work well when the images are not stable enough, when there are bumps on the road the accuracy decreases.

Literature Review

Paper: Object Detection and Recognition using one stage improved YOLOv3

- The primary concern for using this detector is speed more than the accuracy
- The running speed is significantly increased which is approximately 442% faster
- This paper does not extend object localization and recognition from static pictures to a video containing the dynamic sequence of images
- Provided the comparisons of different models

Dataset Description

- **nuScenes dataset**
  - Developed by the nuTonomy team at MIT
  - 3D annotations of 23 classes
  - 1600 x 900 resulations
  - More than 50,000 images
  - Data collected from Boston and Singapore

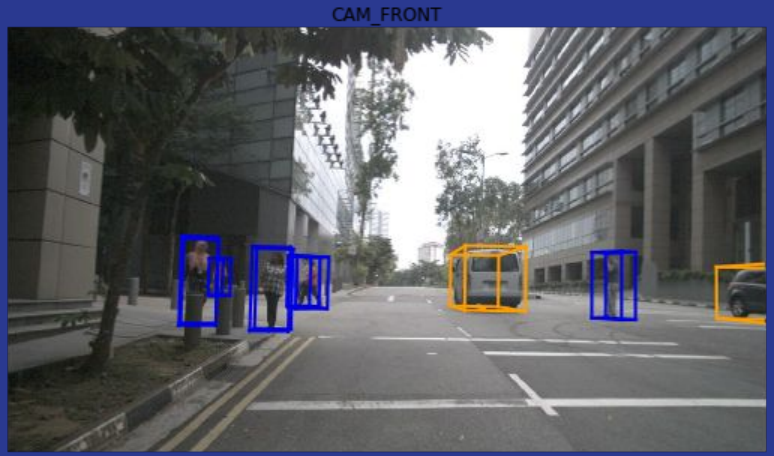
human.pedestrian.adult
human.pedestrian.child
human.pedestrian.wheelchair
human.pedestrian.stroller
human.pedestrian.personal_mobility
human.pedestrian.police_officer
human.pedestrian.construction_worker
animal
vehicle.car
vehicle.motorcycle
vehicle.bicycle
vehicle.bus.bendy
vehicle.bus.rigid
vehicle.truck
vehicle.construction
vehicle.emergency.ambulance
vehicle.emergency.police
vehicle.trailer
movable_object.barrier
movable_object.trafficcone
movable_object.pushable_pullable
movable_object.debris
static_object.bicycle_rack

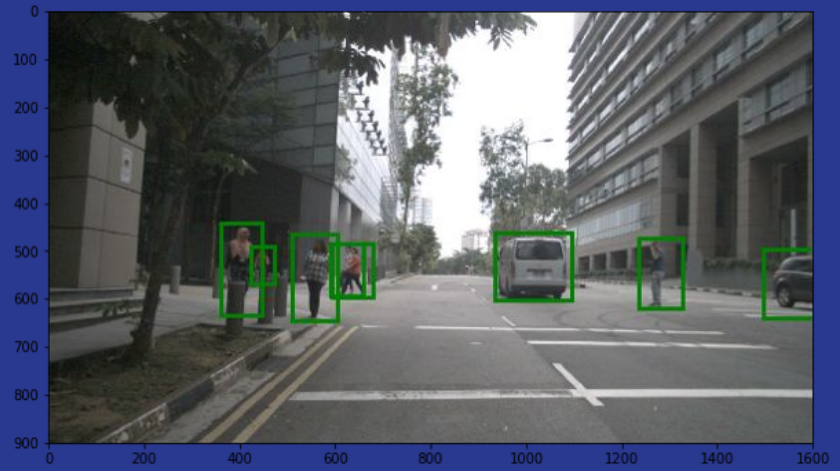
Classes of nuScenes Dataset

Data Preprocessing

- Extracting 2D bounding boxes from corresponding 3D bounding boxes
- Splitting the dataset into 70% for training, 15% for validation and 15% for testing
- Default confidence score threshold is 25%
- Resolution and subdivision of the base model are 416 x 416 and 8 respectively

3D to 2D Bounding Boxes





```python
def threeD_2_twoD(boxsy,intrinsic): #input is a single annotation box
    '''
    given annotation boxes and intrinsic camera matrix
    outputs the 2d bounding box coordinates as a list (all annotations for a particular sample image)
    '''
    corners = boxsy.corners()
    x = corners[0,:]
    y = corners[1,:]
    z = corners[2,:]
    x_y_z = np.array((x,y,z))
    orthographic = np.dot(intrinsic,x_y_z)
    perspective_x = orthographic[0]/orthographic[2]
    perspective_y = orthographic[1]/orthographic[2]
    perspective_z = orthographic[2]/orthographic[2]

    min_x = np.min(perspective_x)
    max_x = np.max(perspective_x)
    min_y = np.min(perspective_y)
    max_y = np.max(perspective_y)



    return min_x,max_x,min_y,max_y
```

3D to 2D Bounding Boxes

Extracting Bounding Boxes

```
def extract_bounding_box(i,camera_name): #give a single sample number and camera name

    '''
    input sample number i, camera name
    outputs min x, max x, min y max y, width and height of bounding box in image coordinates
    2d bounding box
    options for camera name : CAM_FRONT, CAM_FRONT_RIGHT, CAM_FRONT_LEFT, CAM_BACK, CAM_BACK_RIGHT,CAM_BACK_LEFT
    '''

    nusc.sample[i] #one image

    camera_token = nusc.sample[i]['data']['%s' %camera_name] #one camera, get the camera token

    path, boxes, anns, intrinsic_matrix = get_sample_data(nusc,'%s' %camera_token) #gets data for one image

    x_min, x_max,y_min,y_max,width,height, objects_detected,orig_objects_detected = all_3d_to_2d(boxes,anns, intrinsic_matrix)

    return x_min, x_max, y_min, y_max, width, height, path, boxes,intrinsic_matrix, objects_detected,orig_objects_detected
```

Dataset Description

- **Roboflow dataset**
  - Annotations of 10 classes
  - Various resulations
  - More than 9,000 images

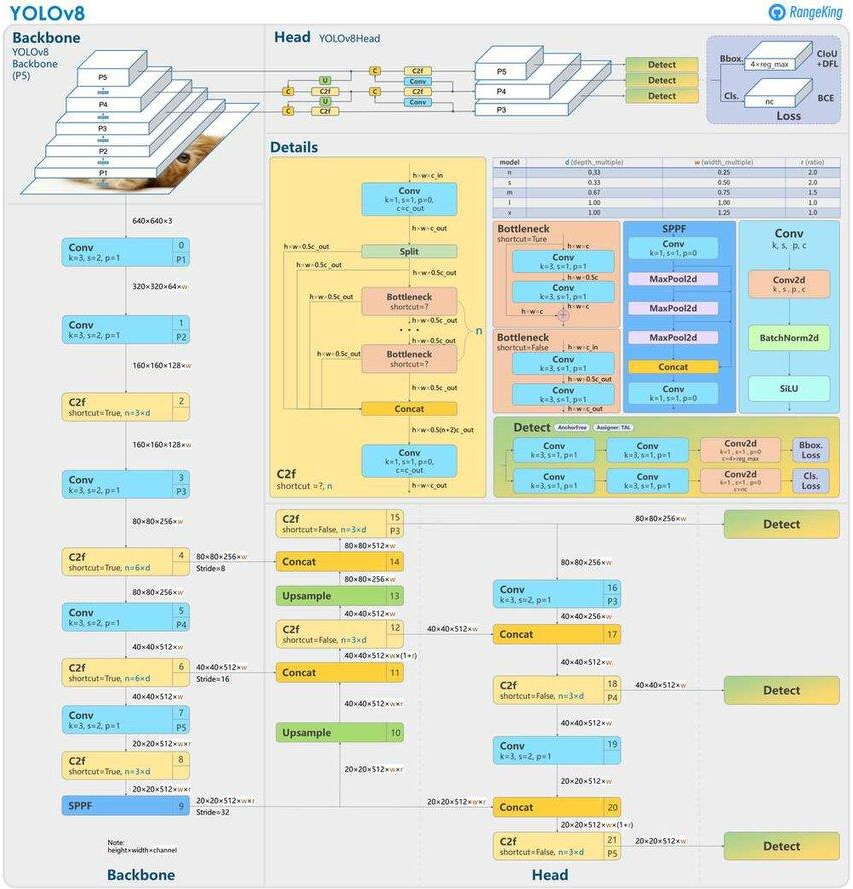
Data Preprocessing

- Adam optimizer
- Blur(p=0.01, blur_limit=(3, 7))
- MedianBlur(p=0.01, blur_limit=(3, 7))
- ToGray(p=0.01)
- CLAHE(p=0.01, clip_limit=(1, 4.0)
- Splitting the dataset into 85% for training, 10% for validation and 5% for testing

Classes of Roboflow dataset

Model for Project

- **YOLOv8**
  - Faster compared to other versions
  - Less computational requirements
  - Real-time implementation
  - Still developing



Structure of YOLOv8

Thank You