Semantic segmentation for autonomous driving

By : ABID Mohanned
BEJAOUI Bedis
TLILI Ahmed

Coach : Mr TLILI Fethi

TABLE OF CONTENTS

01
Problem

The objective of this project is to develop deep learning algorithms to understand the environment of an autonomous vehicle and allow it to navigate safely.

All classes and their labels

Value	Tag
0	None
1	Buildings
2	Fences
3	Other
4	Pedestrians
5	Poles
6	RoadLines
7	Roads
8	Sidewalks
9	Vegetation
10	Vehicles
11	Walls
12	TrafficSigns

02 Technology stack

Technologies used

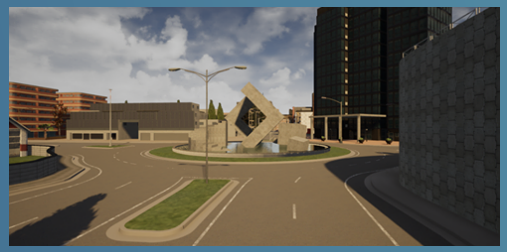
03
Data

Data

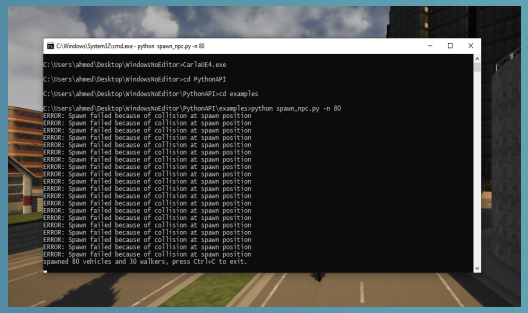
Carla

To generate learning data and carry out autonomous driving tests, we used the CARLA driving simulator

Adding vehicles and pedestrians



Empty Road

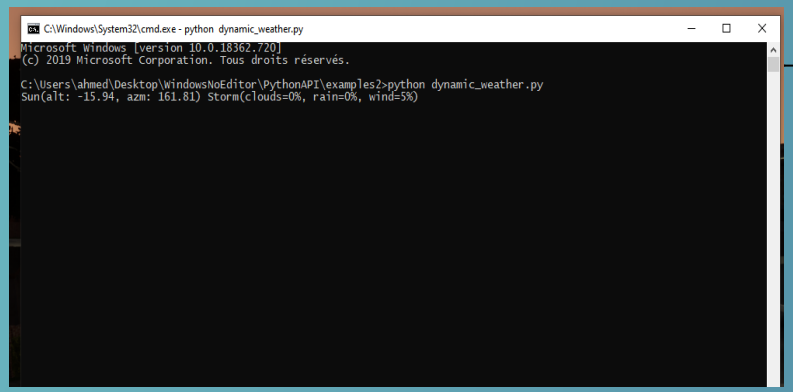


Instructions to add vehicles and pedestrians

Road filled with vehicles

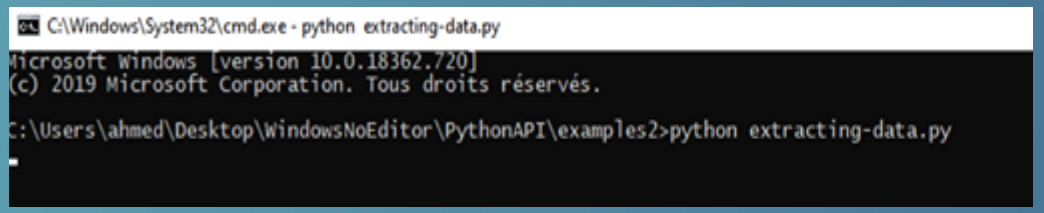
The Weather

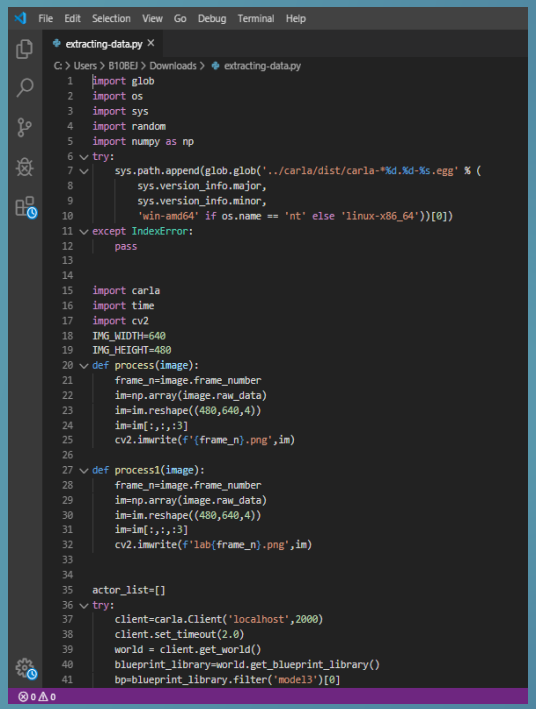
This instruction generate a dynamic weather



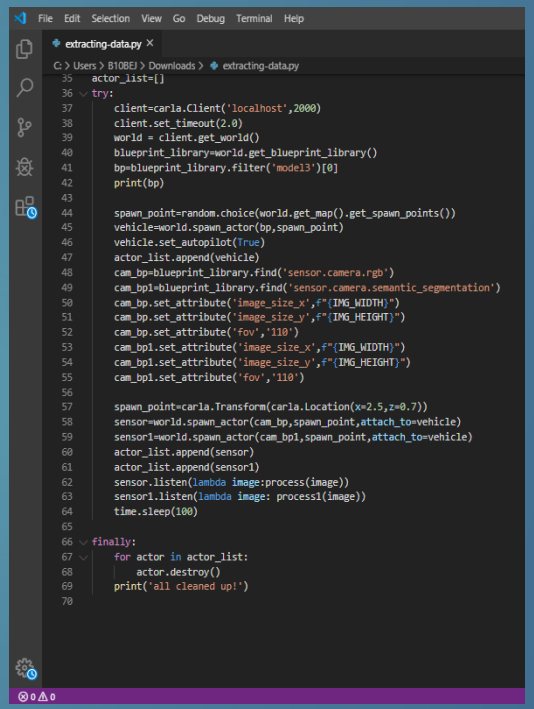
Generate and export our Data

Instruction and python script



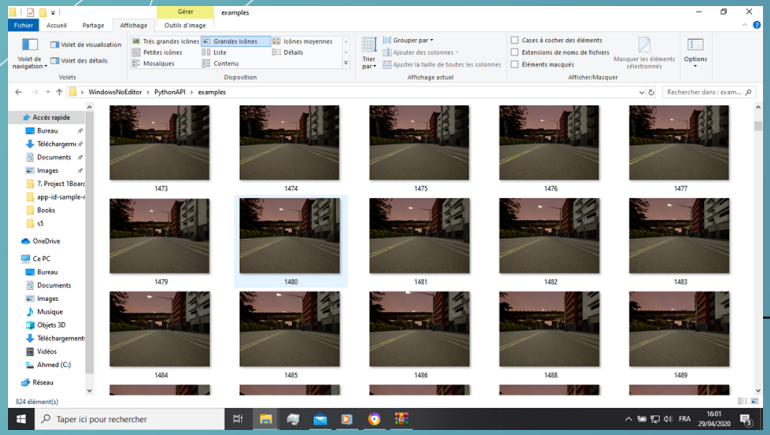


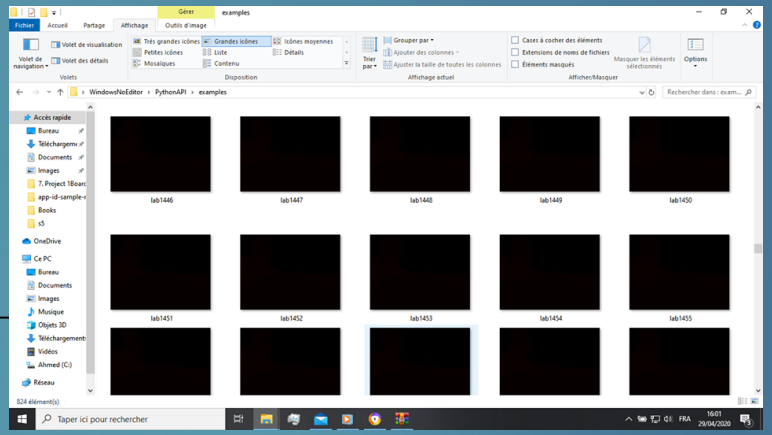
```python
import glob
import os
import sys
import random
import numpy as np
try:
    sys.path.append(glob.glob('../carla/dist/carla-*%d.%d-%s.egg' % (
        sys.version_info.major,
        sys.version_info.minor,
        'win-amd64' if os.name == 'nt' else 'linux-x86_64'))[0])
except IndexError:
    pass


import carla
import time
import cv2
IMG_WIDTH=640
IMG_HEIGHT=480
def process(image):
    frame_n=image.frame_number
    im=np.array(image.raw_data)
    im=im.reshape((480,640,4))
    im=im[:,:,:3]
    cv2.imwrite(f'{frame_n}.png',im)

def process1(image):
    frame_n=image.frame_number
    im=np.array(image.raw_data)
    im=im.reshape((480,640,4))
    im=im[:,:,:3]
    cv2.imwrite(f'lab{frame_n}.png',im)


actor_list=[]
try:
    client=carla.Client('localhost',2000)
    client.set_timeout(2.0)
    world = client.get_world()
    blueprint_library=world.get_blueprint_library()
    bp=blueprint_library.filter('model3')[0]
```



```python
actor_list=[]
try:
    client=carla.Client('localhost',2000)
    client.set_timeout(2.0)
    world = client.get_world()
    blueprint_library=world.get_blueprint_library()
    bp=blueprint_library.filter('model3')[0]
    print(bp)

    spawn_point=random.choice(world.get_map().get_spawn_points())
    vehicle=world.spawn_actor(bp,spawn_point)
    vehicle.set_autopilot(True)
    actor_list.append(vehicle)
    cam_bp=blueprint_library.find('sensor.camera.rgb')
    cam_bp1=blueprint_library.find('sensor.camera.semantic_segmentation')
    cam_bp.set_attribute('image_size_x',f"{IMG_WIDTH}")
    cam_bp.set_attribute('image_size_y',f"{IMG_HEIGHT}")
    cam_bp.set_attribute('fov','110')
    cam_bp1.set_attribute('image_size_x',f"{IMG_WIDTH}")
    cam_bp1.set_attribute('image_size_y',f"{IMG_HEIGHT}")
    cam_bp1.set_attribute('fov','110')

    spawn_point=carla.Transform(carla.Location(x=2.5,z=0.7))
    sensor=world.spawn_actor(cam_bp,spawn_point,attach_to=vehicle)
    sensor1=world.spawn_actor(cam_bp1,spawn_point,attach_to=vehicle)
    actor_list.append(sensor)
    actor_list.append(sensor1)
    sensor.listen(lambda image:process(image))
    sensor1.listen(lambda image: process1(image))
    time.sleep(100)

finally:
    for actor in actor_list:
        actor.destroy()
    print('all cleaned up!')
```

Dataset





Images

Labels

Data Partitioning

80% Training

10% Validation

10% Test

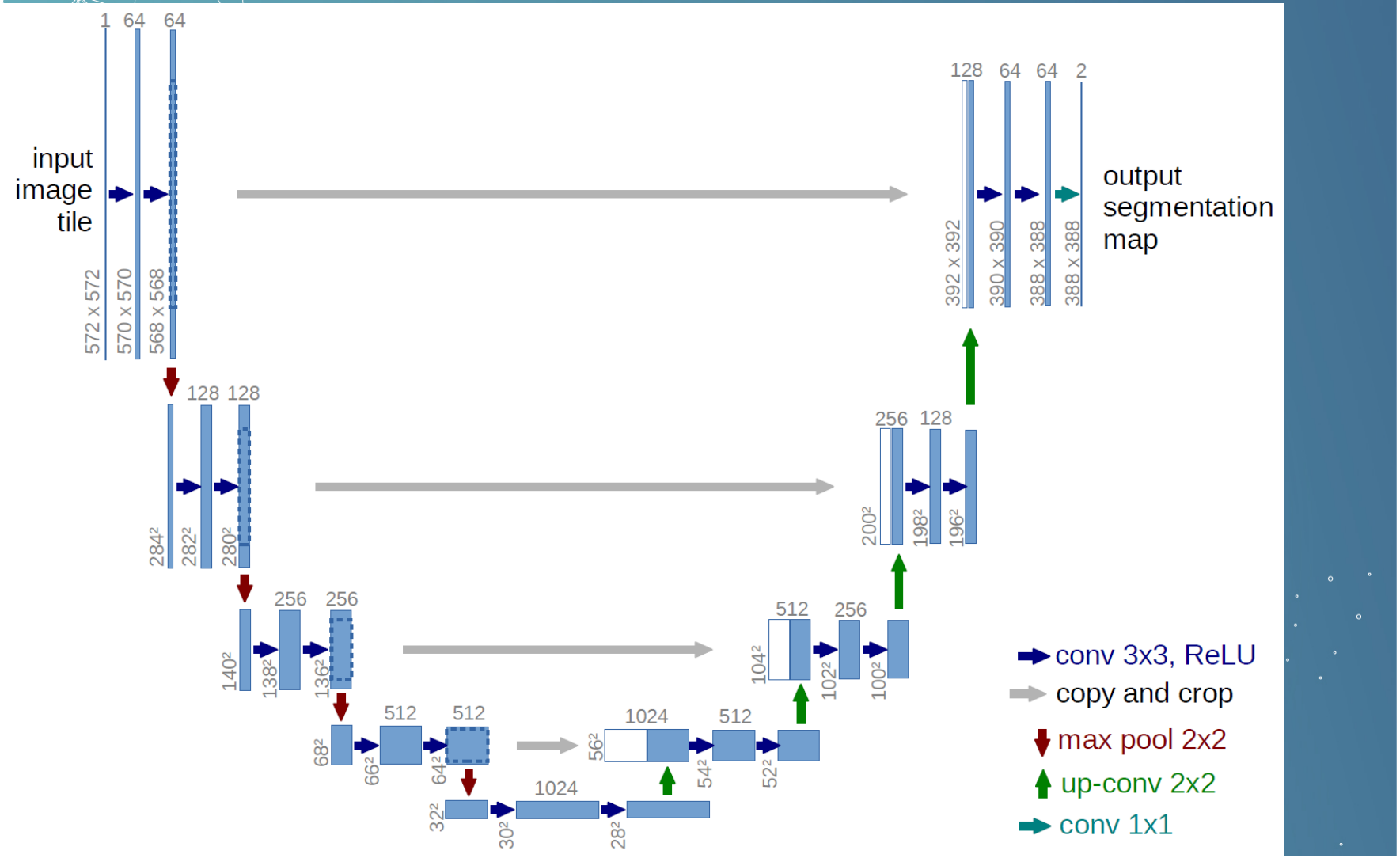
04

Model architecture

U-NET Architecture

U-Net is a convolutional neural network that was developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg

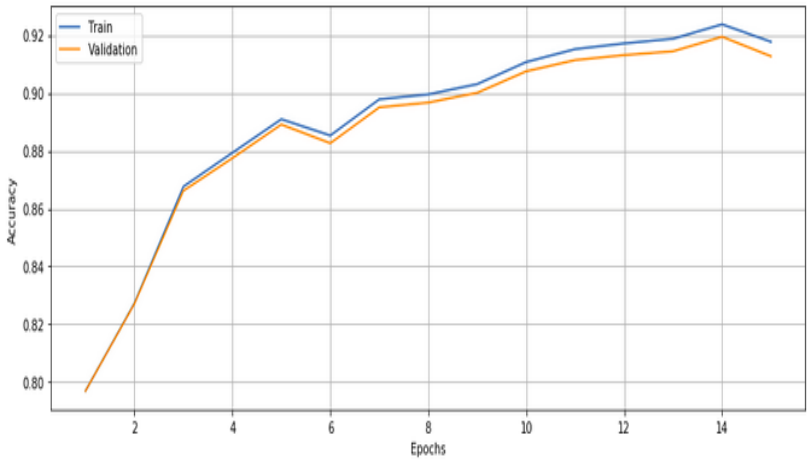
he network is based on the fully convolutional network and its architecture was modified and extended to work with fewer training images and to yield more precise segmentations

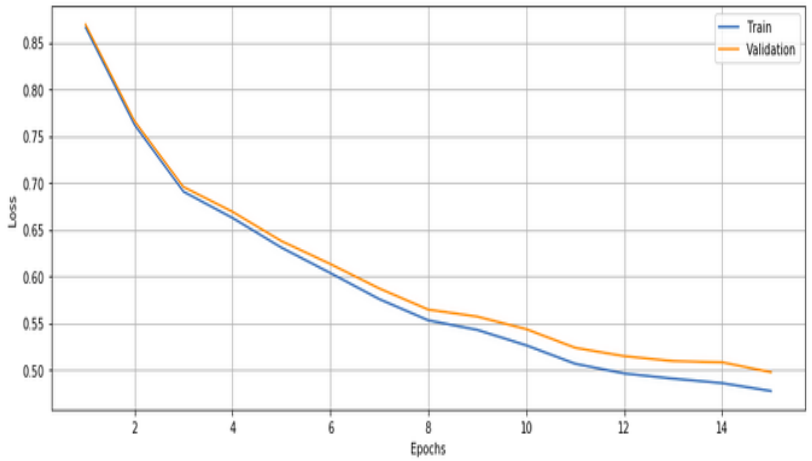

input image tile
572 x 572
570 x 570
568 x 568
1
64
64
128
128
284²
282²
280²
256
256
140²
138²
136²
512
512
68²
66²
64²
1024
32²
30²
1024
512
56²
54²
512
256
104²
102²
100²
256
128
200²
198²
196²
128
64
64
2
392 x 392
390 x 390
388 x 388
388 x 388
output segmentation map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

05

Model benchmarks

Model benchmarks





training_time=4hours
Epochs=15 batch_size=16
img_size=224
learning_rate=0.001

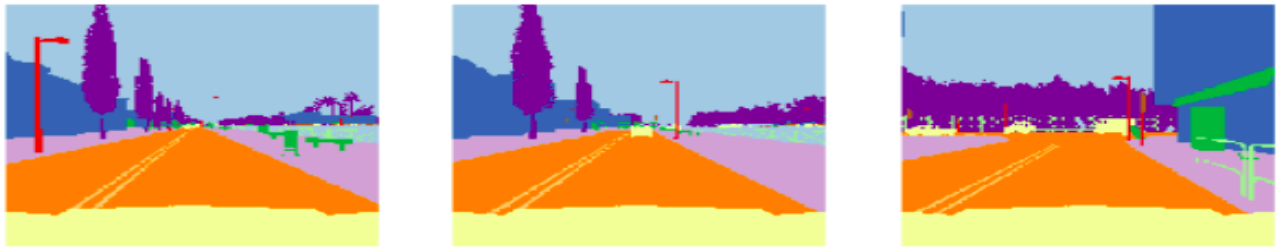
06

Model results

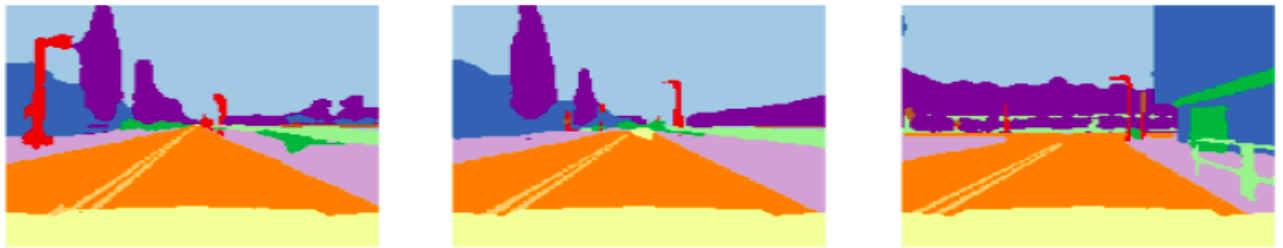
Model results

RGB IMAGES

GROUND TRUTH IMAGES



PREDICTED IMAGES



07
CONCLUSION

CONCLUSION

Our version of U-NET managed to achieve good results on our test set with an accuracy of 92%
but with low resolution (224,224) due to lack of computional resources

You can find the source code here:
https://github.com/mohaned-abid/segmentation-for-autonomous-driving