1 sudo apt install nvidia-driver-510 nvidia-dkms-510

2 sudo apt-get update -y

3 sudo apt install nvidia-driver-510 nvidia-dkms-510

4 hwinfo --gfxcard --short

5 sudo lshw -C display

6 nvidia-smi

7 sudo reboot

8 nvidia-smi

9 sudo apt install python3-pip python3-dev

10 sudo -H pip3 install --upgrade pip

11 sudo -H pip3 install virtualenv

12 df -h

13 ls

14 pwd

15 mkdir jupyter\_dir

16 cd jupyter\_dir

17 virtualenv jupyter\_env

18 source jupyter\_env/bin/activate

19 pip3 install jupyter

20 jupyter notebook --generate-config

21 vi /home/paperspace/.jupyter/jupyter\_notebook\_config.py

22 jupyter notebook password

23 jupyter notebook --no-browser

ssh

cd jupyter\_dir/

source jupyter\_env/bin/activate

jupyter notebook --no-browser

**Abstract**

Autonomous vehicles are the trend nowadays and many of manufacturers introduce innovations to boast their intelligent vehicles. With the advent of science, many assistive technologies emerged to help the driver during long drives like hands-free driving, cruise control, auto adaptive headlights, etc. Artificial intelligence also takes its position in assisting the commutation to be safer and more effortless with technologies like self-driving, lane detection, collision detection, etc. Although collision detection has been there for a while in the automotive industry for many famous manufacturers like Mercedes, BMW, Audi, and Tesla generic danger prediction and warning have not been explored much. This research is proposing to predict threats in advance so that the driver is aware before it is too late to make a decision. The idea here is to use Ultralytics Yolov5 for quick object detection which is a Convolutional Neural Network designed by Ultralytics. Annotated road surface data RDD2020 was used for performing the tests. The data contains classifications like potholes, debris, and cracks which are used to train the model and then later used to warn the driver. Yolov5 has different models, and a comparison of the models was performed. An analysis of the hyperparameters was done to fine-tune the detection results.

Keywords: road damage; convolutional neural networks; semantic segmentation; Yolov5

**Introduction**

**Motivation**

Commutation has always been a need of man to achieve his goals, to be productive, to socialize, to engage in business and many more. Various modes of travel also have been invented, yet the road remains the most widely used. The safety of vehicles is also regulated by the governments across the world and manufacturers comply with guidelines and standards Innovations like seat belts and driver assistance technologies have greatly improved road safety. However, human errors are still prone, and the time is right to discuss how we can reduce manual errors by timely warning using intelligent driver-assistive technologies. During a span of 3 years in US around 200,000 accidents occurred due to road debris which could have been avoided if a proper warning system was available in the vehicle.

**Objective**

There have been various attempts of solving this problem using different sensors, radars, etc. The challenges are the detections should be real-time and should be efficient enough to detect irrespective of the size of the threat being detected. A method well discussed is semantic segmentation which classifies objects at the pixel level.

**Scope**

**Methods**

Yolov5 will be trained with different annotated data sets for road surface detection and road signs and used for predictions and generating warnings. During this research, the performance of Yolov5 is evaluated by gathering various metrics and compared.

**Object Detection**

**Convolutional Neural Networks**

Artificial Neural Networks are used to simulate the decision process of biological neurons using computational networks. They have multiple layers (at least three). All the neurons in a layer are connected to every neuron in the adjacent layers. This is comparable to weighted graphs in which nodes are neurons and connections between the neurons are the directed edges of the graph. Depending upon the prediction validations the weights are adjusted to make further predictions more accurate. This process is called learning and is repeated several times until the errors in the prediction is very small or acceptable after a certain number of iterations called epochs.

**Image Segmentation**

**Image Classification**

Image classification is a classic machine learning problem. [HERE] Although it is very easy for humans to recognize objects like handwriting, warning signs, people, and things, it is a complex process to make computers recognize objects from mere videos or images. There are a variety of techniques that have been applied to solve this problem, like Harr Cascade Classifier, Template Matching, State Vector Machines, etc. However, Convolution Neural Networks (CNNs) generally have been very effective in such complex tasks involving unseen images. The image selected for CNNs processing will be analysed based on its height, width, and channel (grayscale or RGB). The height and width of the image together represent the number of pixels of the image. Convolutional layers in the network process the image and generate a smaller set of features which will be passed to the next layer. Pooling layers perform the down sampling along the spatial dimensions. The fully connected layer does the classification by assigning scores by assessing the features extracted from the previous layer.

Image classification algorithms determine the class of the image while image detection algorithms draw a bounding box around the object found in the image. There can be many bounding boxes based on the number of objects identified in the image. To cater to this, the neural network needs a variable-size output layer. This is not in accordance with the normal convolutional neural network. The R-CNN algorithm by Girshick et al. [HERE] extracts 2000 regions from the image which is fed and then proposed regions are resized and passed into a CNN. The CNN would then classify the input regions. A modified form of this algorithm is the Fast R-CNN.

**YOLO**

The algorithms mentioned in the previous sections have the drawback that they consume too much time for pre-processing. In the case of R-CNN, it defines regions, and then the regions are sent for classification. YOLO (You only look once) on the contrary inputs an image split into SxS default regions and all the regions are processed at the same time. The regions with a confidence level above a certain threshold will be candidates to identify objects within them.

**Releases**

YOLO was first released in May 2016 by Joseph Redmon. In 2017, a better version named YOLO 9000 was released. The most popular and stable version named YOLOv3 was released in 2018 with t paper “YOLOv3: An Incremental Improvement”. “[YOLOv3: An Incremental Improvement](https://arxiv.org/pdf/1804.02767.pdf)”. In April 2020, Alexey Bochkoviskiy introduced YOLOv4: Optimal Speed and Accuracy of Object Detection. YOLOv4 outperformed YOLOv3 by a high margin.

Chart, line chart

Description automatically generated

Source: [YOLOv4 paper](https://arxiv.org/pdf/2004.10934.pdf).

On 9th June 2020, Glenn Jocher, an unofficial author, released YOLOv5 based on PyTorch with exceptional improvements. YOLOv5 is so far the best compared to the previous versions.

Chart

Description automatically generated

source: <https://github.com/ultralytics/yolov5>

**YOLOv5 Model comparison**

Calendar

Description automatically generated

**Source:** https://github.com/ultralytics/yolov5#user-content-pretrained-checkpoints

**Custom Object Detection with Yolov5**

Training YOLOv5 can be defined in the following steps:

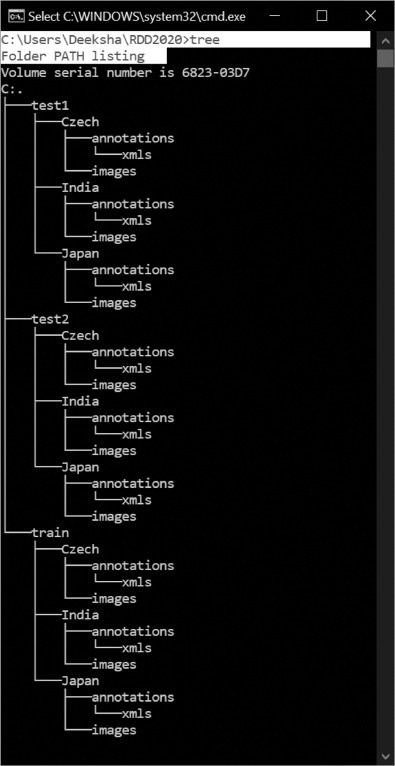
1. Prepare the Dataset
2. Setup the Environment
3. Configuring the parameters in the model
4. Train
5. Evaluate

**Prepare the Dataset**

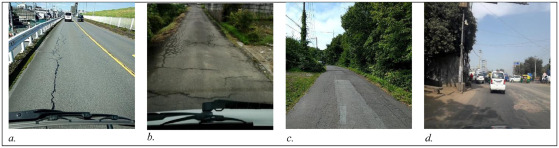
Initially, MS COCO dataset was used to do the validations after the environment was set up. The documentation for YOLOv5 gives pretrained checkpoints given in the above figure based on validation against MS COCO val2017 [http://cocodataset.org/] dataset. Therefore the initial runs were executed to validate the environment with MS COCO dataset. Since YOLOv5 has different models namely, YOLOv5x, YOLOv5n6, YOLOv5m6, etc each of those models was validated with the MS COCO Dataset. YOLO documentation gives results by training each model to 300 epochs.

Since our intent is to predict threats and hazards during driving, we have categorized two types of data that can be used to train YOLOv5. The first one is RDD2020 [https://www.sciencedirect.com/science/article/pii/S2352340921004170] which can be used for training YOLO on road conditions. The dataset contains 26,336 road images collected from India, Japan, and the Czech Republic with more than 31000 instances of road damage. There are four classes mainly longitudinal cracks D00, Transverse Cracks D10, Alligator Cracks D20, and Potholes.

The directory structure for the data is given in the following diagram.



Source: [https://www.sciencedirect.com/science/article/pii/S2352340921004170]



Sample images for road damage categories considered in the data. a. Longitudinal Crack (D00) b. Transverse Crack (D10) c. Alligator Crack(D20) d. Pothole(D40).

The Second category dataset is a small one called LISA Dataset[https://makeml.app/datasets/road-signs]. This dataset contains four image classes.

1. Traffic Light
2. Stop
3. Speed Limit
4. Crosswalk

It contains only 877 images and is a very small one but suits the purpose of this research.

Both the datasets had to be converted into a YOLO-supported format to ensure the training and detection process. The online tool named Roboflow was used to annotate and generate augmented images to train both categories, Road Surface and Sign Board warnings. The generated annotated dataset contains more than 12k images which is a very less quantity but sufficient for the purpose of this research.

**Setting Up the Environment**

The environment consists of:

1. Server hosted by Paperspace.com datacentre in California [8Core CPU, 30 GB Memory, 8 GB GPU, Storage 100GB]
2. PyTorch
3. Jupyter Notebook
4. Wandb.ai [For tracking the metrics during training and testing]

**YOLO Architecture and Working**

Yolov5 architecture consists of three parts:

1. Backbone: CSPDarknet
2. Neck: PANet
3. Head: Yolo Layer

The data is initially fed into the CSPDarknet for feature extraction. The extracted features are then fed into PANet for feature fusion. The last stage Yolo layer detects yields the results (class, score, location, size)

Diagram

Description automatically generated

[https://www.researchgate.net/figure/The-network-architecture-of-Yolov5-It-consists-of-three-parts-1-Backbone-CSPDarknet\_fig1\_349299852]

**Literature review**

During this research, we will be doing a study of the following Computer vision classification models which are Open Source:

1. Vision Transformer
2. OpenAI Clip
3. Resnet34
4. EfficientNet

Also,

1. Mask RCNN
2. Faster RCNN
3. Detectron2
4. Unet
5. Deeplab

Object Detection for Autonomous Vehicle using Single Camera with YOLOv4 and Mapping Algorithm https://ieeexplore.ieee.org/document/9702764

Light-weighted vehicle detection network based on improved YOLOv3-tiny https://journals.sagepub.com/doi/full/10.1177/15501329221080665

Integrated real-time object detection for self-driving vehicles https://ieeexplore.ieee.org/document/8342340

Object Detection in Self Driving Cars Using Deep Learning https://ieeexplore.ieee.org/document/9633965

A Comparison on Instance Segmentation Models <https://ieeexplore.ieee.org/document/9708272>

Road Damage Detection and Classification with Detectron2 and Faster R-CNNhttps://ieeexplore.ieee.org/document/9378027

DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs <https://ieeexplore.ieee.org/document/7913730>

Infrared Image Semantic Segmentation Based on Improved DeepLab and Residual Networkhttps://ieeexplore.ieee.org/document/8530003

**Discussion**

**Confusion Matrix**

https://towardsdatascience.com/confusion-matrix-and-object-detection-f0cbcb634157

https://towardsdatascience.com/confusion-matrix-and-object-detection-f0cbcb634157

YOLO notebooks:

https://colab.research.google.com/github/ultralytics/yolov5/blob/master/tutorial.ipynb#scrollTo=-WPvRbS5Swl6

https://colab.research.google.com/github/ultralytics/yolov5/blob/master/tutorial.ipynb

YOLO file structure

https://chowdera.com/2021/12/202112311417265974.html

Vehicle Detection and Speed Tracking Based on Object Detection

https://link.springer.com/chapter/10.1007/978-981-15-8377-3\_34?noAccess=true