Identification of Universal Challenges in Visual Perception for Autonomous Vehicles

Suruchi Gupta (19233027)
School of Computer Science
National University of Ireland, Galway

Supervisors
Prof. Michael Madden

In partial fulfillment of the requirements for the degree of
MSc in Computer Science (Artificial Intelligence)
September 07, 2020

DECLARATION

I, Suruchi Gupta, do hereby declare that this thesis entitled Identification of Universal Challenges in Visual Perception for Autonomous Vehicles is a bonafide record of research work done by me for the award of MSc in Computer Science (Artificial Intelligence) from National University of Ireland, Galway. It has not been previously submitted, in part or whole, to any university or institution for any degree, diploma, or other qualification.

Signature: _____

Acknowledgement

Firstly, I wanted to express my sincere gratitude to my supervisor, Prof. Michael Madden, who has been an excellent mentor in guiding the project journey. Thank you for your continuous supervision, encouragement and patience that has assisted me in successfully completing this research. I appreciate your time and support in drawing up experimental plan, configuring the models and technical discussions.

I would also like to thank my program director Dr Michael Schukat for the endless support and motivation throughout the course. A heartfelt thanks to all the artists who brighten-up the surroundings with colours. Your figment of imagination perked up this project and made it even more enjoyable.

Finally, I extend a special word of thanks for my family and friends for being my strength during my masters' journey, especially in the challenging times during the pandemic.

Abstract

The recent advances in the field of Artificial Intelligence have immense potential for the realization of self-driving applications. The past few decades have seen massive technological developments to enhance the vision in Autonomous vehicles (1). However, studies demonstrate that even state-of-the-art models can confidently misclassify some natural scenarios (2). These scenarios can raise numerous concerns; most significantly, their use in autonomous vehicles poses a substantial threat in transportation safety as object detection in the autonomous vehicles should be accurate, robust, and real-time (3).

This project aims to create a dataset of real-life images that can potentially mislead autonomous vehicles, test the images against the existing state-of-the-art models to measure the models' performance, and identify the situations that might be a source of risk to transportation safety. We will create a repository of hand-collected, publicly available images to capture the scenarios which are easily distinguishable by humans but might be ambiguous for the object detection models in autonomous vehicles. These images are then labeled and grouped in broad scenarios like traffic signs, vehicle art/textures, etc. After that, the images are tested on existing, state-of-the-art object detection models to determine the scenarios that can perturb an autonomous vehicle. The findings will be used to measure performance, identify trends, and further refine the dataset.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The aim of this research is to create a dataset of images, which can potentially obstruct the performance of self-driving applications. Further, these images will be tested on object detection algorithms to study their impact on autonomous vehicles. These images represent the naturally existing scenarios in the physical world and are not synthetically generated.

1.1 Background

An Autonomous Vehicle (AV) or self-driving car perceives its environment using sensors such as radars, sonar, GPS, etc. and uses an advanced control system to identify the appropriate navigation path (1). Here, an autonomous vehicle relies on the field of Computer Vision to interpret and understand the visual components around them.

The attempts to provide computers with an understanding of visual components around them dates back to the 1960s (17). Computer Vision initially used algorithms for extracting edges, identifying lines, and using the small structures to identify the image elements in the image (18). However, Convolutional Neural

Networks (CNNs) have revolutionized computer vision in recent years. Some interesting approaches for object detection can be witnessed in the ImageNet Large Scale Visual Recognition Challenge (19). The challenge results demonstrate that the model approaches human performance and lags by a small factor of 1.7% (19).

With significant advancements in the field of computer vision, numerous research studies, conversely, show that small perturbations or anomalies in the data can adversely influence the results of the object detection algorithms (2) (8) (7). These perturbations, also known as the adversarial examples, can occur naturally (2) or be user-constructed (8), making the object detection algorithms vulnerable to attacks. For instance, an autonomous vehicle can compromise the passenger's safety or crash if it identifies a painting of a road and vehicles on the wall and confuses it for a real road with vehicles. Hence, the identification and analysis of these scenarios are necessary for road safety. Moreover, these examples can also be used to design a robust autonomous vehicle, immune to similar attacks.

1.2 Problem Statement

This research aims to curate a dataset of naturally occurring images that might act as adversarial examples and highlight limitations of current computer vision systems that might compromise self-driving applications' performance and safety. The repository comprises publicly available, hand-collected images that capture the use cases encountered by the autonomous vehicles classified under the categories: traffic signs, vehicle art/textures, art in surrounding/murals, parking spaces, and on-road scenarios. Furthermore, the images are tested using state-of-the-art algorithms to assess their effects on autonomous vehicles.

The research questions explored in this project are:

(i) What scenarios can act as adversarial examples for the self-driving applications?

(ii) How can these scenarios affect the performance of autonomous vehicles?

1.3 Methodology

The methodology for this project can be broadly divided into:

(i) Gathering images for scenarios that can potentially influence the performance of Autonomous vehicles

(ii) Evaluating the images mentioned above by analysing them with state-of-the-art models

(iii) Summarizing the outcomes for the state-of-the-art model on the collected images

The research involves an iterative approach for steps i) and ii). Initially, we collect some sample images for each category, configure the state-of-the-art object detection models. These sample images are tested with the Object Detection models to refine the process of image collection and to understand what scenario may or may not be challenging for autonomous vehicles.

Part i) involves gathering of images using the following steps:

(a) Identification of potentially confusing scenarios for each category

(b) Collection of images for each of the scenarios

(c) Refining the curated set by mining, similar scenarios on the web

Part ii) evaluates the collected samples by:

(a) Configuring the state-of-the-art object detection or semantic segmentation model

(b) Run the inference for the collected images

(c) Analyse the results to assist the collection of images for similar scenarios

In the end, we summarize the results in step iii) using:

(a) Manually enlist the ground truth details for the images collected

(b) Implementing evaluation metrics and summarizing individual results

(c) Using the summary generated to infer the overall outcome of the project

1.4 Thesis Layout

The thesis is composed of five chapters, briefly construed as:

- Chapter 1 introduces the project, the research questions explored, and the methodology followed for the project.
- Chapter 2 reviews the fundamentals of Neural Networks with the concepts of Object Detection and Semantic Segmentation in Computer Vision. Further, it studies the existing architecture for Autonomous Vehicles, Datasets available for assisting autonomous vehicles, adversarial examples, and their impacts on Object Detection models. Finally, it examines the state-of-the-art models (YOLOv3, Faster R-CNN, and DeepLabv3 model) and reflects on Evaluation Metrics used further in the project.

- Chapter 3 presents details regarding the categories and scenarios studied under each category to explain how they pose adversarial examples for autonomous vehicles.
- Chapter 4 outlines the software and the hardware specifications on which the images are tested. It also explains the configuration details and results for the models used.
- Chapter 5 summarises the project results and suggests some opportunities for future work on this project.

Chapter 2

Literature Review

This chapter provides an outline of Neural Networks and how they are employed in Computer Vision. Later, we review the research conducted in Computer Vision, Autonomous Vehicles, and Adversarial examples with concepts relevant to this project.

Section 2.1 introduces the fundamentals of Neural Networks and Convolutional Neural Networks. Section 2.2 outlines the Computer Vision tasks, namely Object Detection and Semantic Segmentation. Subsequently, Sections 2.3 and 2.4 provide details regarding Autonomous Vehicle architecture and the datasets for self-driving tasks. In section 2.5, we introduce Adversarial Examples and the research on them so far. Section 2.6 examines the architectures of the state-of-the-art models employed, and finally, section 2.7 studies the Evaluation Metrics used in this project.

2.1 Introduction to Neural Networks

An Artificial Neural Network, loosely inspired by the biological brain, is an interconnected group of nodes that distributes information processing tasks and

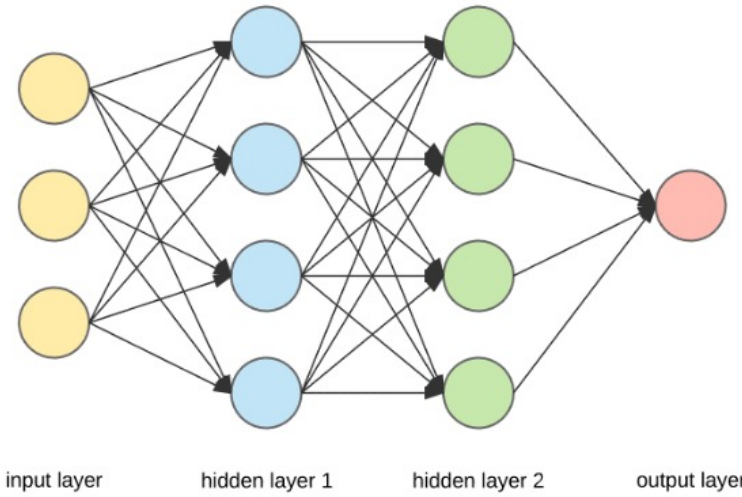

Figure 2.1: A Sample Neural Network Architecture from (4)

creates patterns to enable decision making (20). For sample neural network with two hidden layers in Fig. 2.1 by Yang (4), each node is an artificial neuron. Additionally, the arrow represents a connection between input, hidden, and output nodes. For the case, each node in a layer is connected to all the nodes in the previous layer, also called Fully-Connected Feed Forward Neural Network.

Each node in the network has weights associated with it; the input to the node is multiplied with these weights and passed to a non-linear activation function that helps in learning the representation of the given classes (4). Some examples of commonly used activation functions are sigmoid, tanh, and ReLU. The network is trained using chain rule where the error from the output node is backpropagated to the input nodes, and the weights are updated iteratively to reduce the error across the nodes (21).

The nodes in the Neural Networks can be connected in numerous ways to address different challenges. However, Convolutional Neural Networks with multiple hidden layer architecture are extensively used for image analysis and are used as a base for modern algorithms (13) (22).

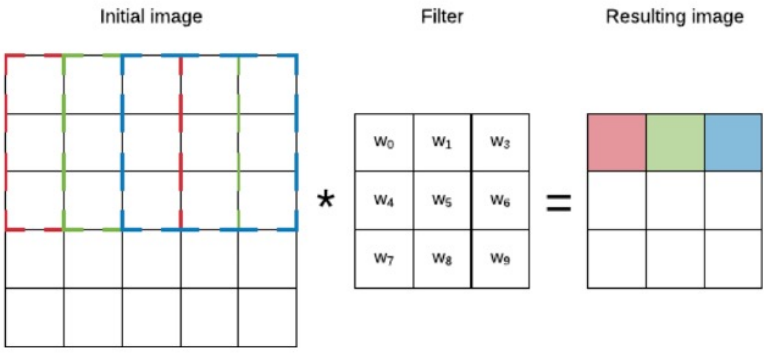

Figure 2.2: First three Convolution steps example from (5)

2.1.1 Convolutional Neural Network

For image analysis tasks, it is logical to imagine that each pixel in the image relates more to the pixels around it than those far from it. This local context feature in the images is the basis to exploit Convolutional Neural Network (CNN) for image analysis (23).

CNNs use a convolution matrix (illustrated as 'filter' in Fig. 2.2 from Padarian et al. (5)) to identify features in the image using shared weights for the connected nodes. Multiple convolution layers augmented with pooling layers (used to reduce the dimension by clustering the outputs from the previous layer into one node in the current layer (24)) are used to identify complex features in the image. The convolution layers are followed by a fully-connected layer to flatten the features and classify them into available class labels.

2.2 Computer Vision

Computer Vision deals with the training of computer systems to interpret and understand the visual components around them. Modern-day Computer Vision applications employ the concepts of Artificial Intelligence on the digital images

to recognize the objects in the surroundings (18).

This process of analysing digital images to extract meaningful information from them is known as Image Analysis. The Image Analysis components are highly application-dependent; however, some of the essential components can be defined as image acquisition, pre-processing, feature extraction, detection/segmentation, high-level analysis, and decision-making (25).

This research will collect the images and test them for Object Detection and Semantic segmentation (discussed at length further in this section) and use the results to identify adversarial scenarios for autonomous vehicles.

2.2.1 Object Detection

Effective outcomes in Computer Vision heavily rely on the safe and reliable execution of underlying models; one essential model in this pipeline is Object Detection. It is the process of identifying instances or objects in digital images. An object detection task typically returns the objects present in the image, enclosed in boxes that contain them, sample input, and output of the Object Detection task presented in Fig. 2.3.

The classical Object Detection pipeline used a series of steps like pre-processing, the region of interest extraction, object classification, and verification, or refinement (1). The pre-processing involves camera calibration, image rectification, exposure, and gain adjustment. Region of interest (ROI) can then be extracted using a sliding window that shifts at different scales over the image to identify objects with varying sizes. Further, a search through the extracted ROIs provides us with the objects identified in the image. As an exhaustive search is expensive, we can alternately use a selective search that uses image segments to identify the approximate locations of interest efficiently. These extracted regions are then used for classification and verification. As there can be numerous image

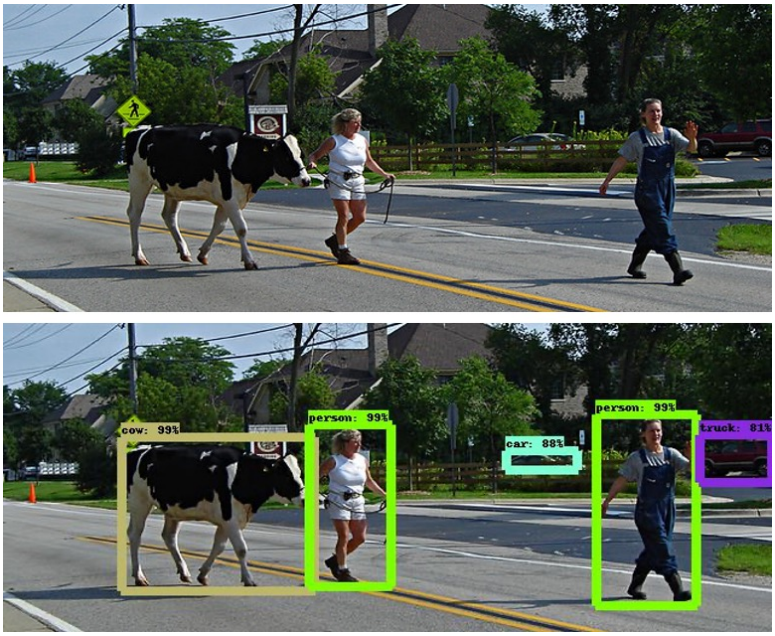

Figure 2.3: Sample Object Detection task

Sample image (top) from COCO dataset ran through Faster R-CNN model (bottom) for Object Detection task.

regions for processing, the algorithm must be quick in discarding the noise or background regions. Multiple approaches use this method and show promising results for object detection using SVMs with Histogram of Orientation(HOG) (26) (27).

The preceding steps were effectively reduced to one trained model using deep neural networks for object detection. CNNs use a sliding window on a shallow network to identify the objects' precise features and return the class label with a bounding box. There are multiple architectures identified to efficiently perform object detection with CNNs (11) (22) (12) (13), moreover, most of the state-of-the-art approaches use CNNs in the underlying architecture to identify and extract features from the images. The current research shall use two pre-trained models for object detection viz; Faster R-CNN and YOLO(You Only Look Once).

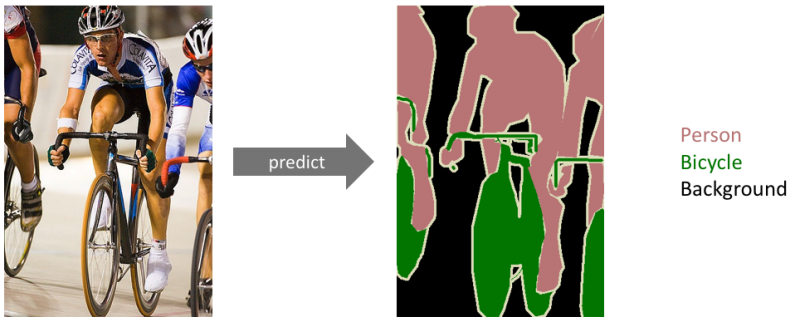

Figure 2.4: Semantic Segmentation Example

Sample image (left) from VOC2012 dataset ran through Semantic Segmentation model (right) from (29)

2.2.2 Semantic Segmentation

As noted earlier, the goal of Object Detection is to find a bounding box that contains the entire object. Semantic segmentation takes a step ahead and aims to classify each pixel in the image into one of the class labels (28). It finds applications in tasks like image search engines, autonomous vehicles, medical imaging analysis. As for autonomous vehicles, it aims to understand the scene to assist in making control decisions (1) by segmenting the image in various regions. An instance of a semantic segmentation task from the Cityscapes dataset provided in Fig. 2.4 from PASCAL VOC (29) demonstrates that each class label is assigned a specific colour (for example, the person as red, car as blue, and so on) and each pixel in the image is assigned a specific class label.

A simple solution to the semantic segmentation task is Image Thresholding. Here, a grayscale image is converted to a binary image using a threshold to classify different contrasts in the image (30). Some other traditional semantic segmentation methods include K-means clustering, watershed algorithm, etc. (31). However, the use of Deep Learning techniques has been a turning point for semantic segmentation as well.

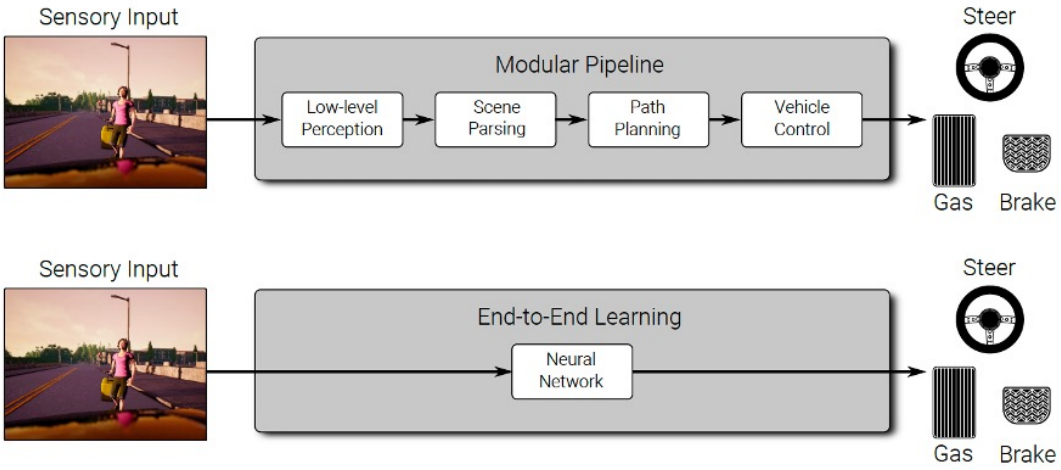

Figure 2.5: Autonomous Vehicle Architectures
Modular Pipeline (top) and End-to-End learning model (bottom) from (1)

The early deep learning techniques use Fully Convoluted Network (FCN) for encoding and then decode the feature map for segmentation results (31). The encoding process involves multiple convolutions along with pooling to reduce the image size and extract the essential properties from the image. Finally, as the semantic segmentation task outputs a map with the same size as the input image (1), a decoder is used for upsampling the features obtained to higher resolution and hence, achieve dense predictions. Advanced techniques like SegNet, DeepLab, etc. use FCN as a basis to handle Semantic Segmentation tasks(31). We will discuss in-detail and use the DeepLab version 3 model to perform semantic segmentation.

2.3 Autonomous Vehicles Architecture

The internal architecture of Autonomous Vehicles can be broadly classified under either of two, modular pipelines or end-to-end learning approach. Fig. 2.5 from Janai et al. (1) provides a general overview for both approaches.

2.3.1 Modular Pipeline

The modular pipeline breaks down the complex functions into small modules, each of which can be developed, trained, and tested independently. An approach to modularization can constitute low-level perception, scene parsing, path planning, and vehicle control (1).

Low-level perception and scene parsing is typically accomplished using the DNNs, whereas state machines, search algorithms, and control models are used for path planning and vehicle control (32). As this approach breaks down each decision into a specific set of intermediate modules, it is much easier to comprehend and predict the outcome for a scenario.

The Modular Pipelines are advantageous as they are comparatively easy to integrate, the development of the modules can be parallelized, and humans can interpret the results of the intermediate modules for failure assessment (1). However, one must also consider that the intermediate modules might not be optimal as each modules' outcome might not consider the final driving task's actual goal, like travel time, safety, and comfort.

2.3.2 End-to-End Learning Approach

The End-to-End Learning Approach, instead of Modular pipelines, use a single trained model to learn a policy by observing the surroundings and choosing an appropriate action (1). This approach learns a strategy by mapping the sensory inputs, such as front-facing camera images, to driving actions such as steering angle. The intermediate steps taken by the model to arrive at a decision are not known.

The model parameters are learned either by observing the environment, as in Reinforcement Learning, or by replicating a teacher's behaviour, as in Imitation Learning. However, Reinforcement Learning is challenged by reward shaping (it is

difficult to define the rewards for complex tasks) and credit assignment problem (assigning each decision step a part of the credit for a completed task) (33). Additionally, Reinforcement Learning is slow and not suitable for safety-critical simulations. As for Imitation Learning, it experiences overfitting and does not generalize well to the unseen cases; hence, compromise the vehicle's safety for the new scenarios.

Finally, the End-to-End Learning-based Models are hard to interpret as we cannot predict what sensory input is resulting in a particular decision (1). Since the driving decisions are directly mapped to sensory inputs, they show the "black box" characteristic of DNNs for the developer or user.

2.4 Datasets for Autonomous Vehicles

With multiple automakers working in the realization of autonomous vehicles, there are multiple proposals for the datasets specific to the Autonomous Vehicles. The first publicly available dataset was published by Geiger et al. (34) called the KITTI dataset. The cameras of an autonomous driving platform were used to acquire data for this dataset. It has around 13,000 images for multiple scenarios like Road, City, Residential, Campus, etc. The KITTI dataset has been refined and is known as one of the standard datasets for self-driving applications but is often only used for evaluation purposes attributing to its limited size (1).

Cordts et al. (35) published the Cityscapes dataset with pixel-level semantic labelling for images related to urban scenarios. The dataset comprises of 25,000 images (sample images in Fig. 2.6) for 50 different cities, 5000 pixel-level annotations (precise annotations from website (36) in left image of Fig. 2.6) and 20,000 weak annotations (blur object boundaries from website (36) presented on right in Fig. 2.6) . This dataset, however better than the KITTI dataset in terms of

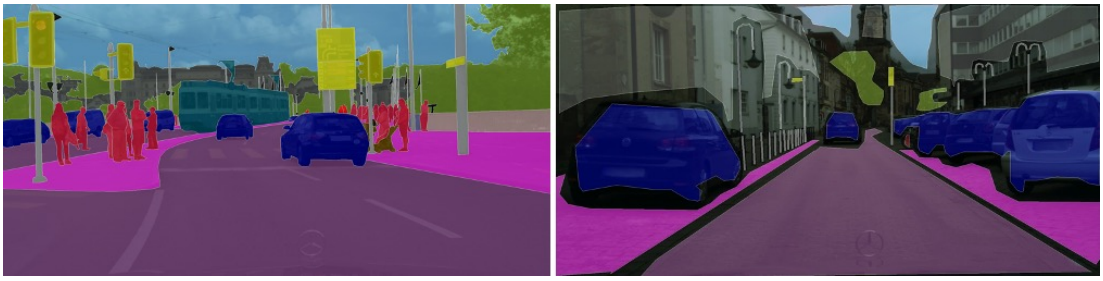

Figure 2.6: Sample images from Cityscapes dataset

Images showing pixel-level(left) and coarse annotations(right) available at (36)

detailed annotations, only captures urban scenarios.

Multiple companies working on autonomous vehicles have started making their datasets publicly available. For instance, the ApolloScape dataset (37) provides labelled 140,000 images of the street view for lane detection, car detection, semantic segmentation, etc. The dataset is curated meticulously to enable performance evaluation across different times of day and weather conditions (1).

For the current research, as the aim is to identify the adversarial scenarios, a more generic dataset – COCO dataset, is used as the basis of testing the collected images. COCO (Common Objects in Context) (38) is a dataset by Microsoft with 328,000 images. It has 80 labels of the commonly available objects in the surroundings that can easily be identified by any preschool student (sample from website (6) presented in Fig. 2.7). The results for most of the latest research are generally provided on the COCO dataset, hence, used as the base for this research.

Figure 2.7: Sample images from COCO dataset from (6)

2.5 Adversarial Examples for Object Detection

With the increasing use of DNNs for efficient image processing, there has been a lot of analysis of how these models can be fooled or attacked. Multiple experiments conducted in recent years show that even small perturbations in the data can significantly decrease the model's performance, also known as Adversarial examples. These perturbations can either be naturally-occurring unseen scenarios (Natural Adversarial examples) or human-constructed to motivate the model to make a mistake.

The DNNs are highly non-linear and sometimes called "black-boxes", as the non-linearity makes the transformation from input to output difficult/impossible to explain. While some experiments studied the effect of adding noise to the existing image data to mislead the model (8) (7), others studied the impact of the natural adversarial examples on the performance of state-of-the-art models (2).

Szegedy et al. (7) used a pre-trained network and derived a perturbation specific to the image by altering the weights to minimise the cost function. When overlapped with the original image, this noise goes untraced by the human eye

Figure 2.8: Adversarial Examples for AlexNet dataset from (7)

but highly hampers the performance of the model. An example from the paper illustrated in Fig. 2.8 depicts the original images on the left of both panels, the noise generated to increase the prediction error on the middle, and the adversarial example generated by overlapping the earlier mentioned, on the right. The images on the left of the column of the panels are classified correctly by the model. However, all the images on the panel's right column are classified as into 'Ostrich' class label.

Similarly, Nguyen et al. (8) tested the models on images with atypical-generated noise, which is beyond recognition for humans. However, the models classified the noise as objects with high confidence values. Fig. 2.9 shows an instance from work by Nguyen et al. (8), the left panel has the original images, and the right panel presents the images generated by the evolutionary algorithms, optimised to produce high-confidence predictions on ImageNet dataset. These findings have encouraged work in identifying the possible attacks that can mislead or fool even the state-of-the-art models.

Similarly, Hendrycks et al. (2) collected a set of natural images for seen classes to the model and observed that some of the natural un-altered images

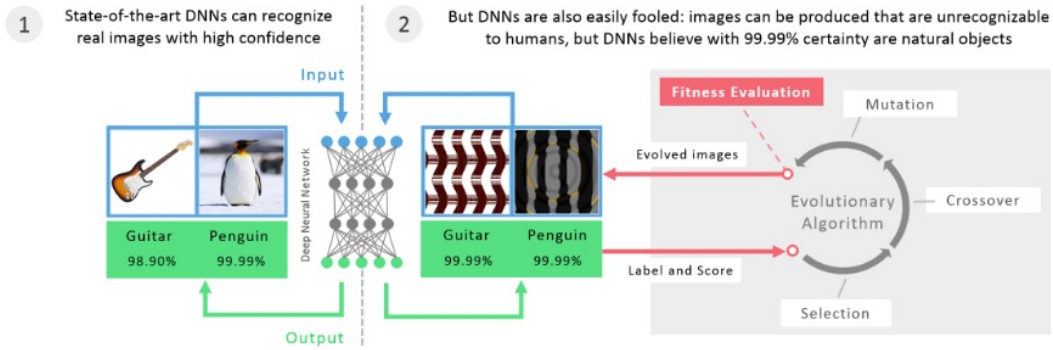

Figure 2.9: Fooling DNNs using Evolutionary algorithms from (8)

could also mislead the model significantly. The images in Fig. 2.10 from the research show that different scenarios such as texture, shape, background, etc. can provoke the DNNs to misclassify the image instances. They further extended the experiment by adding a set of images with unseen classes for the model and expected low confidence scores or high anomalies. Conversely, the model predicted the classifications with high confidence scores, raising severe concerns for the model's reliability for un-seen examples.

Figure 2.10: Natural Adversarial Examples from (2)

With the identification of the issue in DNNs, to be fooled by the adversarial

examples, multiple experiments also acknowledge the use of adversarial examples to improve the existing models and harness these examples to build models resistant to the adversarial attacks (39) (40). While Xie et al. (39) have used adversarial examples as a sample space while training the model to prevent overfitting and improving the overall performance of the model. Madry et al. (40), on the other hand, have laid out optimization techniques to handle "the first-order adversary" and building adversary robustness in the models for accurate classification results. Hence, the images collected in this research can not only be used to test the autonomous vehicles in the edge-case scenarios, but they can also be used for building better and robust autonomous vehicles immune to the adversarial attacks for added safety.

2.6 Current State-of-the-art Architectures

2.6.1 YOLOv3

You Only Look Once (YOLO), suggested by Redmon et al. (22), is an approach to Object Detection that identifies bounding boxes and the class probabilities for the objects simultaneously. It makes predictions using a single network evaluation system and promises a 100x fast response compared to Faster R-CNN (41). Hence, it is a good choice for real-time Object detection. Additionally, it makes fewer background errors and successfully learns the generic features of the object.

The introduction of Residual Neural Networks or ResNet (9) by He et al. enabled efficient training of models with even a thousand hidden layers. ResNet uses skip connections, or shortcuts, to jump one or more layers. The skip action in a residual block (presented in Fig. 2.11 from He et al. (9)) not only addresses the issue of vanishing gradient in the Recurrent Neural Networks, but it also amplifies the output of the block by adding the input and leads to an increasing

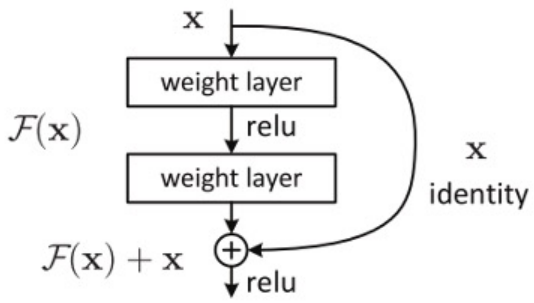

Figure 2.11: A residual block in Residual Neural Network from (9)

effect by passing the input deep into the network. ResNet architecture is found to be very useful for image recognition (9), and hence, it is used in YOLOv3 architecture.

The YOLOv3 (42) is an improvement on existing YOLO architecture, and it uses darknet architecture (open-source neural network framework) as the backbone. This new YOLO architecture is presented in Fig. 2.12 and it uses ResNet and improves by performing predictions at three scales obtained by downsampling the image dimensions by strides of 32, 16, and 8. This helps YOLOv3 for better identification of smaller objects. Moreover, YOLOv3 introduces pre-defined anchor boxes (chosen by performing K-means clustering on the data), 3 for each scale. This helps YOLOv3 to predict ten times more boxes than YOLOv2.

Finally, the model concatenates outputs from all the scales and uses non-max separation to discard the multiple detections of the same object. It then uses the OpenCV library in python to draw the resultant bounding boxes for the input image. YOLOv3, with these new tricks, performs at par with the existing state-of-the-art models and is still considerably faster than the models like RetinaNet, SSD, etc. (42).

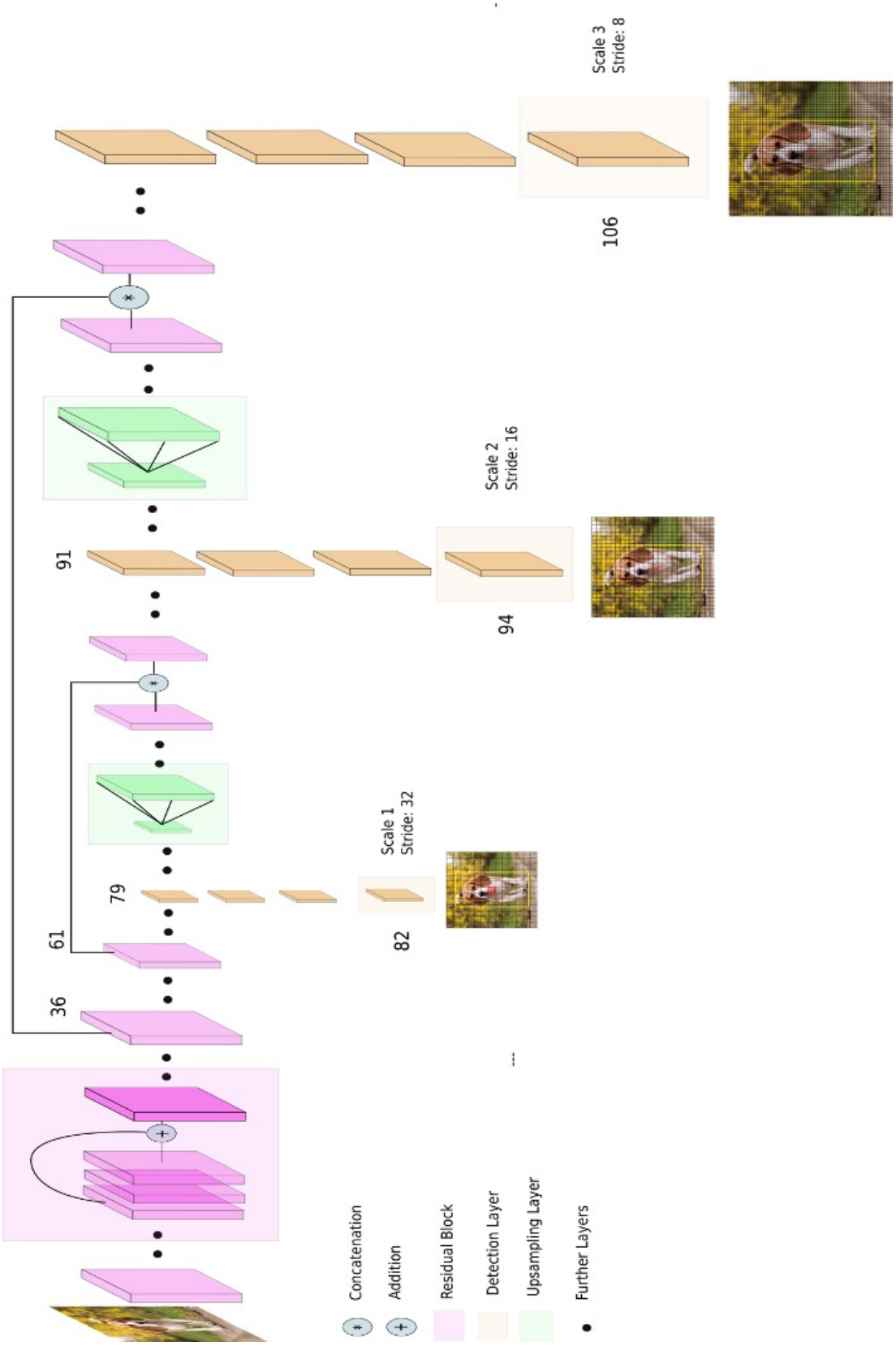

Figure 2.12: YOLOv3 Network Architecture from Ayoosh (10)

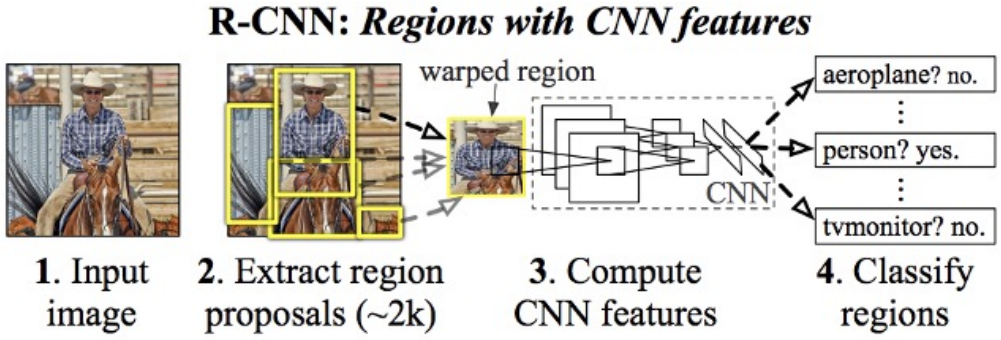

Figure 2.13: Object Detection architecture for R-CNN from (11)

2.6.2 Faster R-CNN

Faster R-CNN architecture has evolved gradually over time from Region-Based Methods. R-CNN by Girshick et al. (11) proposed scanning the input image to identify around 2k region proposals, then use CNNs to identify the features in the regions and classify each region using linear SVM for each class (briefly depicted in Fig. 2.13 from (11)). The region proposals are generated in different sizes by methods such as selective search and are known as the Region of Interest (RoI).

After performing a selective search to identify the RoIs, R-CNN sends all the regions (2k for each image) to the CNN, making it computationally expensive. It takes around 47 secs to run object detection for each image, deeming it unsuitable for real-time object detection (14). Girshick (12) in 2015 proposed Fast R-CNN; this approach sends the entire image to the CNNs and generates feature maps. These feature maps are used to generate the Region of Interest (RoIs), and they are warped in the bounding boxes of fixed sizes using the RoI pooling layer. Finally, a fully connected layer uses softmax function to identify the object in the box (shown in Fig. 2.14 from (12)).

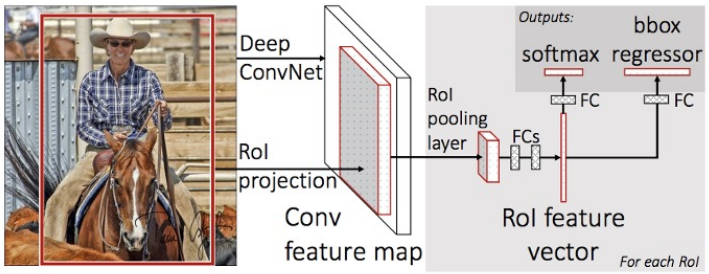

Figure 2.14: Fast R-CNN Object Detection architecture from (12)

Fast R-CNN reduces the execution time to around 2 secs; however, when further analysed the region proposals take up the most time in the object detection process (14). Ren et al. further improved on Fast R-CNN using Region Proposal Network (RPN) (13). RPN is another neural network that replaces the selective search in the Fast R-CNN architecture; this new architecture is also known as Faster R-CNN (presented in Fig. 2.15 - left from (13)). The use of RPN has reduced overall computation and has significantly reduced the overall execution time. The test time for the architectures mentioned above shows improvement in test speed across architectures, depicted in Fig. 2.15 (right) from (14).

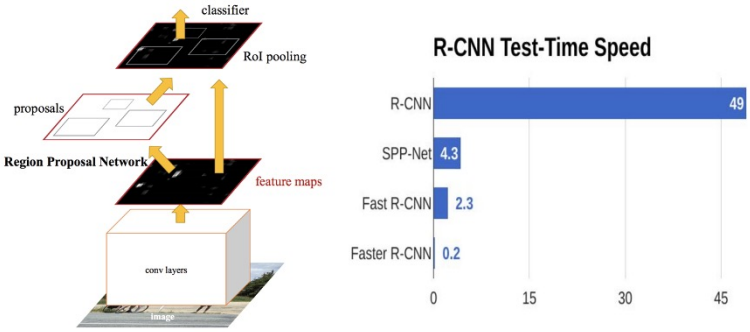

Figure 2.15: Faster R-CNN architecture from (13) and test speed of different architectures from (14)

2.6.3 DeepLabv3 for Semantic Segmentation

DeepLab is a state-of-the-art model for Semantic Segmentation by Google. The major challenge in performing semantic segmentation using Deep CNNs(DCNN) is signal down-sampling leading to fuzzy boundaries and spatial invariance (31). DeepLab (43) achieves the dense predictions using atrous or dilated convolution, i.e., it up-samples the output from the consequent convolution layers resulting in feature maps with a higher sampling rate. Additionally, to address DCNN invariance, it employs fully connected Conditional Random Fields (CRFs) to capture fine details. Altogether, the image is passed through multiple DCNNs, followed by few atrous convolutions to achieve a score-map, which is later up-sampled using bi-linear interpolation. Finally, fully connected CRF further strengthens the segmentation results (model illustration in Fig. 2.16, source Chen et al.(15)).

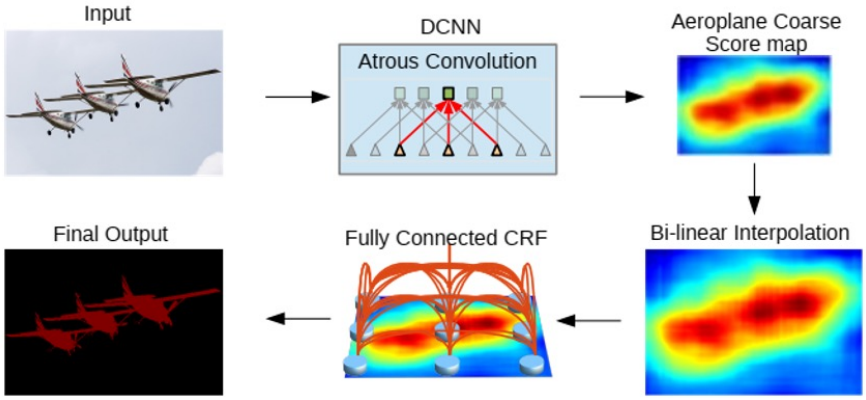

Figure 2.16: DeepLab v1 model flowchart from (15)

The DeepLabv1 model was further enhanced to address the issue of objects of the same class occurring at multiple scales (15) using the Atrous Spatial Pyramid Pooling (ASPP). ASPP applies atrous convolution to the input feature map with different sampling rates and combines it to capture objects as well as useful context at multiple scales in the image (different views depicted using different colours in Fig. 2.17 from Chen et al. (15)). The refined architecture enables the

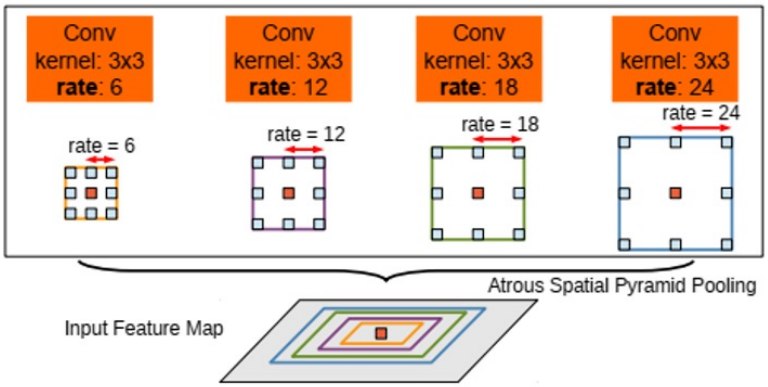

Figure 2.17: Atrous Spatial Pyramid Pooling (ASPP) from (15)

detection of different object scales and improves the accuracy of the model.

DeepLabv3 improves the segmentation boundaries using the atrous convolution with an improved ASPP model that includes batch normalization and image-level features (44). Additionally, CRF is removed from the improved model. The DeepLabv3 architecture (shown in Fig. 2.18 from Chen et al. (16)) uses the backbone network to extract the feature maps from the image and applies atrous convolution in the final layers. Further, the ASPP network classifies each pixel to a class label, and the result is then passed through 1x1 convolution to achieve the final segmented image with actual size (16). The modifications in the model capture long-range context using dense feature maps and enhance the model's reception without reducing the spatial dimension to boost the model's performance.

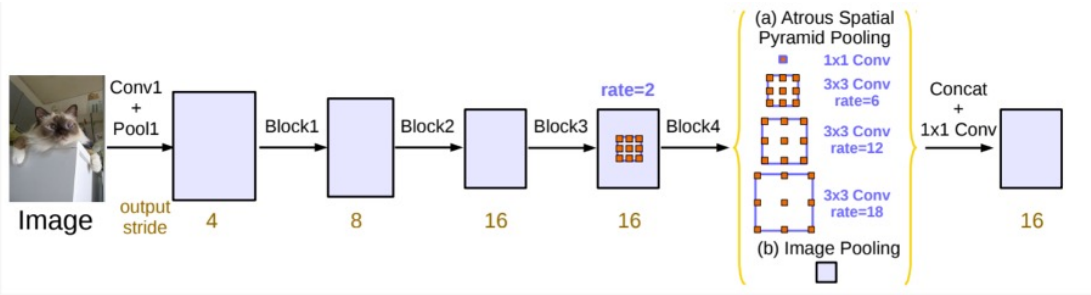

Figure 2.18: DeepLab v3 model architecture from (16)

2.7 Evaluation Metrics

Evaluation metrics are the statistical means to compare the model's outcome with the expected outcome or the Ground truth. These metrics provide a means to compare the performance of different models. There are multiple metrics available to test this experiment; however, we will evaluate it on the following metrics.

2.7.1 Accuracy

Accuracy is measured as a ratio of correct classifications over the total testing instances, statistically given as:

$$Accuracy = \frac{\text{Count of correct classifications}}{\text{Total classification instances}}$$

2.7.2 Confusion Matrix

Confusion matrix is a means of efficiently summarizing the results of a classification problem. Accuracy, as studied by Kocco and Capponi, can be mis-leading for evaluation of imbalanced classes as it favours the majority class (45). Confusion matrix for classification problem is defined as:

		Predicted	
		Class 1	Class 2
Actual	Class 1	True Positive (TP)	False Negative (FN)
	Class 2	False Positive (FP)	True Negative (TN)

Table 2.1: Confusion Matrix for Classification Problem

The specific values in the confusion can be insightful in determining what aspect of the model needs further analysis or training. It can also be used to compute other evaluation metrics like accuracy, precision, and recall.

For the Object Detection task, the evaluation is done using (46):

- True Positive: The object is present in the image and is identified by the model
- False Negative: The object is present in the image but is not detected by the model
- False Positive: The object is not present in the image but is detected by the model
- True Negative: This represents all the undetected parts of the image. As this cannot be determined for object detection task, this is ignored

2.7.3 Precision

Precision is a measure of how close the predictions are to the correct values over all the positive predictions; it is defined as:

$$Precision = \frac{\text{True Positive}}{\text{True Positive+False Positive}}$$

2.7.4 Recall

Recall shows a measure of how far we are in finding all the positive cases and is statistically defined as:

$$Recall = \frac{\text{True Positive}}{\text{True Positive+False Negative}}$$

2.7.5 F1-score

F1-score is the harmonic mean of the precision and recall, it is given as:

$$\text{F1-score} = 2 * \frac{\text{Precision * Recall}}{\text{Precision + Recall}}$$

Advanced metrics like Intersection-over-Union(IoU), Average Precision can be used to evaluate the bounding boxes returned for the images. However, only the number of objects in the images is evaluated for the given research, not the bounding boxes for each object.

Chapter 3

Gathering Data

This chapter introduces the details regarding the dataset created. The images collected are broadly categorised under Street Signs, Vehicle Art and Textures, Art-in-surrounding and Murals, Parking Spaces, and On-road scenarios. Though limited in number, the categories are bundled logically to give the models real-time scenarios from the physical world.

Section 3.1 lays out the prerequisites considered while collecting the images. After that, we explore each category at length. Section 3.2 introduces the scenarios captured in the category Art-in-surrounding and Murals. Section 3.3 provides a detailed review of Vehicle Art and Textures. Subsequently, Section 3.4 explains different On-road scenarios studied in the research. Section 3.5 looks closely at the Street Signs, whereas Section 3.6 deep dives into Parking spaces. Finally, Section 3.7 examines the images tested with Semantic Segmentation, whereas Section 3.8 provides sample Advanced Scenarios which cannot be tested in the scope of this experiment.

3.1 Prerequisites for Image Collection

Data collection is the primary objective of this project, and it aims for this data to be used for further training and evaluation of autonomous driving applications. Hence, the images selected for the analysis in this project adhere to the following properties:

3.1.1 Image Format

All the images used are in JPEG format (i.e., .jpg or .jpeg extensions) as the Object detection model used supports only JPEG format and refrains from identifying other image formats.

3.1.2 Image Type

The images in the dataset are coloured images in RGB format. Each pixel in these images is one of the 256 colours defined in the palette as the combination of Red, Green, and Blue colours.

3.1.3 Image Copyright

The images curated are publicly-available and are labeled for reuse under the creative commons license. The images are free for distribution and are not copyright protected. The source of images is provided in Appendix A

3.1.4 Image Editing

The images used for this project are un-edited for most of the scenarios. However, 3 images in the dataset are cropped to reduce the noise from the image (and to

assist the object detection algorithm). These images are checked in advance that they are labeled for reuse with modification.

3.1.5 Image Size

The images collected are of arbitrary size and are not resized as the models used in the project resize the images to the required dimensions internally. Hence, there is no explicit need to resize the images to specific dimensions.

3.2 Art-in-surrounding and Murals

Art, unrestricted by language, is often used as a universal medium to communicate ideas. Visual art dates back to ancient civilisations, and was used as an effective way of communication without using words. Statistics show, there are around 2000 murals documented in Northern Ireland alone, since the 1970s (47).

Streets and their surroundings, across the world, witness this form of art; some consider it as an alternate means to effective communication, whereas others consider it as serious vandalism. Nevertheless, when introduced to the physical world, an autonomous vehicle must distinguish the work-of-art from reality. Hence, this category aims to identify the artistic creations that exist near roads, and otherwise, the re-creation of these, around the roads, might deteriorate self-driving applications' performance.

3.2.1 Re-creation of a road scenario

Many visual arts are inspired by real-life scenarios or situations and are often recreated on the walls around the road. These images reconstruct a road scenario and potentially trick the object detection models into identifying components like cars, traffic lights, cycles, pedestrians, etc.

Figure 3.1: Sample murals re-creating road scenarios

The model, if it is inefficient in distinguishing these murals with the truth (sample presented in Fig. 3.1), might compromise the safety of fellow road users and the passengers in the vehicle by possibly crashing to the wall.

3.2.2 Identifying safety threat in the surroundings

Some of the murals might contain elements that an autonomous vehicle might identify as a source of threat for the passenger. Some sample scenarios are identifying an accident, wild animals, images depicting natural calamities, etc.

Figure 3.2: Paintings on road depicting safety threat

Though the system should be able to identify these risky scenarios, the false identification of the scenarios (depicted in Fig. 3.2) might lead to the wrong choice of judgment on the part of autonomous vehicles.

3.2.3 Artistic Creation of Road Objects

Besides the painting on the walls, some 3D figures replicate the objects typically found on the road, such as cars, trucks, motorbike, etc. As the object detection algorithm identifies the object's features to judge the presence, these artworks can be confused with the actual objects. A sample of these objects is illustrated in Fig. 3.3.

Figure 3.3: 3 dimensional re-creation of road objects like car and truck

3.3 Vehicle Art and Textures

The vehicles, mostly, come in the standard make and colours. People seldom like their motor vehicles to look different, and so they customise their motor vehicles to have a unique look. These designs, though, are a delight for humans to watch, can baffle the autonomous vehicles. This category contains images of motor vehicles that are different from usual and can incite the autonomous vehicles to make a mistake on the road.

3.3.1 Vehicles disguised as Other Objects

In this category, the images show the motor vehicles camouflaged as different objects such as shoes, telephone, animals (cat, peacock, dragon), flowers, skull, etc. Sample images in Fig. 3.4 show car disguised as cat and motorbike inspired from Star Fleet Academy from famous movie series Star Trek.

Figure 3.4: Motor Vehicles camouflaged as other objects

Interestingly, Houston hosts an Art Car Parade every year to showcase these unique car designs; similar out-of-the-box car design can be found at this link (48).

3.3.2 Vehicles with Textures

Some owners and enthusiasts use texture as a medium to advertise and decorate their motor vehicles. The vehicles are either covered with a specific pattern like grass or patches on the cow, tiger print, etc. or by small assorted patterns to create a unique effect on the automobile body (an instance in Fig. 3.5).

Alternatively, some vehicles can also have a scene painted on their body, such as an anime character or depicting a scene from cartoon books or movies (instances illustrated in Fig. 3.5).

Figure 3.5: Textures to decorate vehicles

3.3.3 Custom Build Unique Vehicles

In addition to these, some motor vehicles are also uniquely designed by the manufacturers as 'Positional goods' to have distinguishing features, such as custom print, solar panels, dual-engine, etc.

The chances for the autonomous vehicles to encounter these limited-edition collectibles are rare (shown in Fig. 3.6), yet proactively studying the effects of these designs is a smart decision. As shown in the sample images, some of these vehicles are a hybrid of different automobiles; for instance, the car that looks similar to a helicopter might hinder the advanced control systems' decision.

Figure 3.6: Custom built motor vehicle

3.4 On-road Scenario

The autonomous vehicles are trained on datasets relevant to the road containing different road objects and scenarios to help the model understand the on-road components. Nevertheless, due to diverse roads and scenarios across the world, an autonomous vehicle cannot possibly be aware of all the components that it might encounter on the road. The images in this category aim to identify the unseen on-road scenarios for the autonomous vehicles that might throw the self-driving applications off balance.

3.4.1 Animals on the Street

As humans expand our habitable land, there are multiple places where encounters with animals on roads is not unusual. Hence, images in this category show

Figure 3.7: Sample images showing animals on road

different species of animal, wild and domestic, wandering across the streets in rural and urban scenarios (sample available in Fig. 3.7).

Unlike previously mentioned scenarios, where the mural was inanimate and drivers in other vehicles are sensible, the animals' reaction for a vehicle can be arbitrary and hence, be difficult for the autonomous vehicle to handle.

3.4.2 Billboards along the Road

Billboards are a common item available alongside almost every road. The major purpose of having them is to quickly catch the driver's attention and advertise the product. Although they are not relevant to an autonomous vehicle, their presence can potentially confuse the system.

The sample images in Fig. 3.8 show some generic billboards, as well as some that are specifically used during the election campaigns. In case an autonomous

Figure 3.8: Examples showing billboards on the road

vehicle identifies the pictures on the billboards as humans, it might consider applying emergency brakes and lead to erratic braking scenarios.

3.4.3 Challenging Driving Scenarios

When employed in the physical world, the autonomous vehicles are subjected to experience the different lighting conditions at different times of the day and varying weather conditions (fog, rain, snow, etc.) throughout the year. Hence, they must be aware of different weather conditions and their resulting impact on the surroundings.

Figure 3.9: Challenging road scenarios for Autonomous Vehicles

The images (illustrated in Fig. 3.9) attempt to capture the changes in the lighting and weather conditions to assess how an autnomous vehicles reacts to them. For instance, the identification of wet and icy roads for careful driving or fog on the roads might compromise the autonomous vehicles' vision.

In addition to the earlier mentioned scenarios, this category has images of regional variations of vehicles (tricycles for public transport, cargo bicycles for delivery, etc.), challenging roads scenarios (such as mountains, valleys, etc.), and images of extremities like accidents so that we can intensely test and fine-tune the autonomous vehicles for these rigorous scenarios.

3.5 Street Signs

A significant component of roads apart from the pedestrians and other vehicles are the street signs. They are used to guide and provide instructions to the road users. This section aims to analyse, in detail, the variations of the street signs across the world. It also identifies exciting modifications to the existing road signs created by some artists.

3.5.1 Art on Street Signs

As Henry David Thoreau mentioned, "The world is but a canvas to our imagination"; some artists have modified the existing road signs elements to create interesting variations.

These variations might be convincing for humans; however, failure to identify these signs on the road (sample images in Fig. 3.10) can have severe implications for autonomous vehicles. As for the sample image on the right in Fig. 3.10, autonomous vehicles can disregard the bump sign on the road and fail to reduce the speed.

Figure 3.10: Street signs with artistic variation on the road

3.5.2 Regional Variations of Street Signs

The street signs, though standard at most places, also have some regional variations. Some regions in Arab, Asia, and Russia, have street signs in the regional language (illustrated in Fig. 3.11).

Figure 3.11: Examples of Street signs in Arabic and Chinese

The autonomous vehicle must be checked and trained for these regional variations before getting deployed, as being unaware of these variations, they might fail to give their optimal performance.

3.5.3 Custom Variations in Traffic Signals

While curating the images relating to street signs, we encountered some custom traffic signs. These traffic signals do not have the standard circular red or green lights; instead, they have some custom figure shown red and green colours. Some of these are applicable for pedestrian crossing, and humans are intelligent to identify them. However, these cases are still added to this category (depicted in Fig. 3.12) to study their effects on an autonomous vehicle's judgment.

Figure 3.12: Custom traffic signals on the road

Furthermore, this category also contains images of a giant mosaic art created using the old discarded street signs. Some elements in the art do not threaten the integrity of the self-driving applications. However, identifying any of the street signs in the art might lead to an unexpected outcome by the vehicle. The COCO dataset only identifies the stop sign; nevertheless, a more comprehensive study with a domain-centered dataset can provide more insights about the effects of these street signs on autonomous vehicles.

3.6 Parking Spaces

As every journey has a start and stopping point to complete the circle of driving-related scenarios, this category covers the parking spaces and its variations. It evaluates autonomous vehicles' performance in the presence of unseen animals or objects in the parking space or a rural parking setting without the standard parking boxes, etc.

3.6.1 Unforeseen objects in Parking Spaces

The presence of unforeseen objects around autonomous vehicles can lead to numerous uncertain scenarios. The images in this category identify miscellaneous objects in the parking spaces, such as animals, shopping carts, etc.

Figure 3.13: Images showing presence of unseen objects in the parking space

Sample images presented in Fig. 3.13 show some of these scenarios. The presence of unidentified objects like shopping carts(left and right-top image in Fig. 3.13) in the parking spaces can lead misjudgment by the autonomous vehicles.

3.6.2 Unconventional Parking Signs and Warnings

Besides the unforeseen objects, there are cases where authorities issue warnings/notices or customised parking / no-parking signs that can puzzle the judgment of the autonomous vehicles.

Figure 3.14: Creative parking signs and warnings in the parking spaces

The sample image on the left in Fig. 3.14 shows one of such unconventional parking signs or notices, the image on the right in Fig. 3.14 shows a warning sign regarding the icy car paths ahead; these warnings can be ignored by the autonomous vehicles and might lead to safety-critical issues.

3.7 Semantic Segmentation

While collecting the art creations along the roads, we found a few images that create an illusion of 3D-space. Hence, to understand how an autonomous vehicle constructs these scenarios, we have subjected these images through a semantic segmentation model.

Figure 3.15: Sample 3D-Illusions tested using Semantic Segmentation task

The images depict multiple scenarios, for instance, a waterfall on the road, a broken road with the presence of wild animals, etc. (sample presented in Fig. 3.15). If unhandled, these adversarial scenarios can be used as a means for adversarial attacks and might compromise the autonomous vehicle's security.

3.8 Advanced Scenarios

The COCO dataset covers only the 80 commonly available objects in the surroundings. It is noteworthy that all the road scenarios cannot be covered in these 80 objects. Hence, the Advanced Scenarios contain the images that contain objects or adversarial scenarios that might be out of sight for a model trained on the COCO dataset. A road-specific dataset on further evaluation might recognize these examples to be adversarial.

3.8.1 Additional Artistic Creations

Section 3.2 details the scenarios where art in the surroundings can potentially mislead autonomous vehicles and might even lead to safety-critical situations. This category contains some art images that do not contain objects from classes in the COCO dataset but can potentially act as adversarial scenarios for self-driving applications.

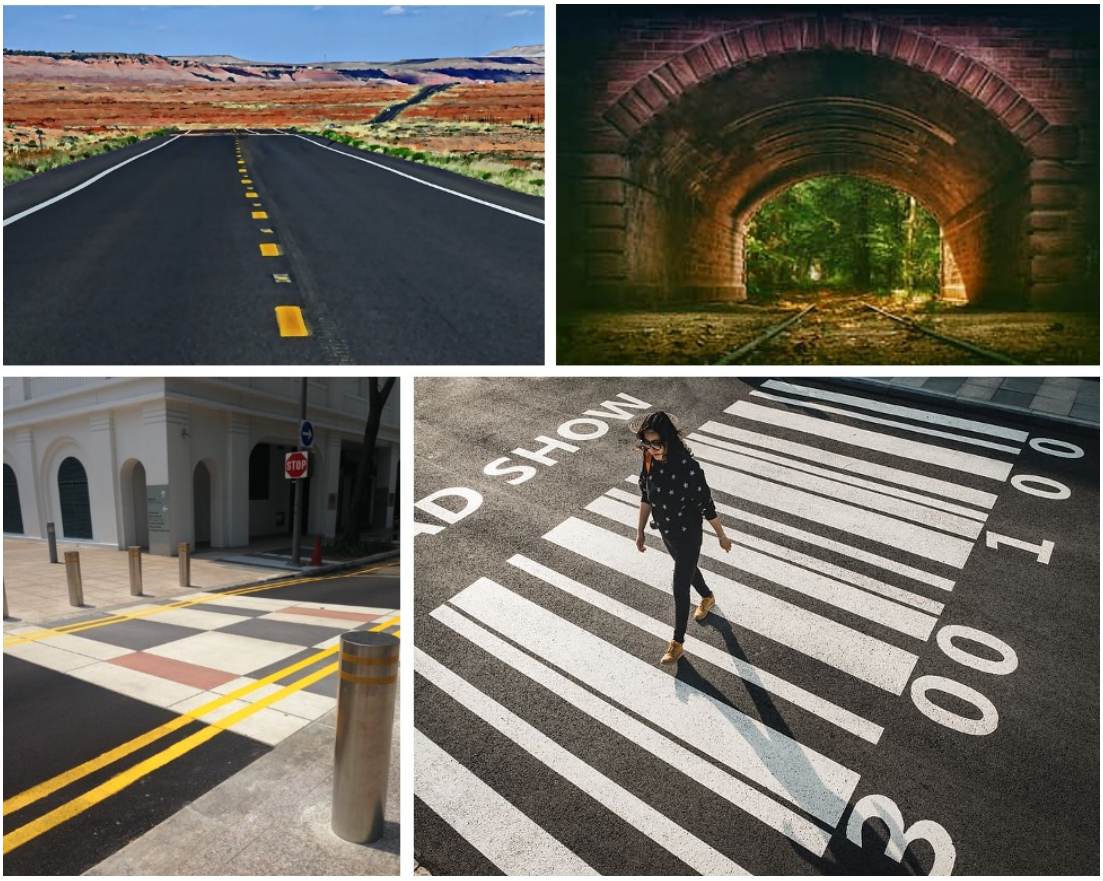

Figure 3.16: Additional scenarios of Art that might act as Adversarial examples

The sample images at the top in Fig. 3.16 show paintings of the road; these might lead the autonomous vehicles to choose these options and compromise the passenger's safety. As for images on the bottom in Fig. 3.16, the images show creative pedestrian crossing, autonomous vehicles that fail to identify these special crossings might put some lives at risk.

3.8.2 Animal Crossings

As we share the habitat and infrastructure with other animal species, different forms of animal crossing signs are commonly used to ensure all the intended users' safety. As these signs can be used in different formats, it can be challenging for

Figure 3.17: Example showing different animal crossing signs

autonomous vehicles to identify the signs and also understand details like caution for a specific distance. This category acknowledges the variety of animal crossing signs across the world and aims to test if the autonomous vehicles can identify them as expected (instances in Fig. 3.17).

3.8.3 Natural Calamities

With the ever-changing seasons and environmental conditions, natural calamities are a risk that cannot be eliminated. These incidents often severely damage the transportation infrastructures around us as well. This category considers the possible situations that might occur in the event of the natural calamities and proposes to examine the response of autonomous vehicles for them.

Figure 3.18: Sample 3D-Illusions tested using Semantic Segmentation task

The images illustrated in Fig. 3.18 capture possible scenarios that show road damaged as a consequence of natural calamity. The instances on the top in Fig. 3.18 show the blockage of road due to a landslide. Whereas, samples on the bottom of the Fig. 3.18 show the outcomes of earthquake and flood, respectively.

3.9 Summary of Dataset

The curated dataset contains a total of 894 images across six categories. The count of images for each category is given in Table 3.1.

Category Name	**Count of Images**
Advanced Scenarios	99
Art-in-surrounding and Murals	226
On-road Scenario	210
Parking spaces	41
Street Signs	99
Vehicle Art and Textures	219
Total	894

Table 3.1: Summary of images in the dataset

The images are available for use in further research at the link and ground truth details are available at link.

The images collected are assigned a unique file name to identify the images with ease. Each image is manually annotated to create ground truth data in the form of an excel file. The ground truth data contains image file name and count of objects for each of the class names present in the COCO dataset. This ground truth data is used for the evaluation of metrics such as accuracy, precision, etc.

The image examples in this chapter provide different challenging scenarios with an intent to train better and test autonomous vehicles. This is done to minimise the risk associated with any possible real-world use case. However, similar natural scenarios can be identified and tested to improve this dataset and the overall safety of the self-driving applications.

Chapter 4

Experimentation and Results

This chapter explains the experimental setup used to test the collected data. Our aim here is to analyse the result of images after running inference through YOLOv3, Faster R-CNN, and DeepLab Semantic Segmentation model, and understand if the use case is an adversarial scenario for an autonomous vehicle or not.

Section 4.1 states the hardware and software configuration used for the project. Section 4.2 and 4.3 discuss the configuration and results for YOLOv3 architecture and Faster R-CNN models respectively. Section 4.4 explains the setup for the DeepLab Semantic Segmentation model.

4.1 Experimental Framework

As discussed in the literature review, this project employs state-of-the-art object detection and semantic segmentation models to test the collected road scenarios. The models used are pre-trained on the COCO dataset to identify 80 common object classes in the surroundings and are not altered during the course of this experiment.

The experiments are conducted on a MacBook Pro running with macOS Catalina version 10.15.6 with 3.1 GHz Dual-Core Intel Core i5 processor and 16 GB memory. The experiment project is divided into two modules, namely, Object Detection and Semantic Segmentation. The Object Detection module comprises all the components required to assist this experiment. The Semantic Segmentation model is an extension used to derive some extra information regarding some of the specific road-related scenarios.

Table 4.1 shows the python scripts implemented to test the Object Detection experiment and evaluate the overall performance. The threshold for the Object Detection models is set as 70%.

Script Name	**Description**
DirectoryReader.py	Contains helper methods to read all the files from /data directory and return a dictionary of folder name as keys and list file paths as values.
YoloObjectDetector.py	Loads the weights for YOLOv3 model, runs inference for each image, saves the images with output class and bounding boxes and records the results for each image to assist evaluation of the metrics.
FrcnnObjectDetector.py	Loads the pre-trained Faster R-CNN model, runs Object Detection for each image, saves the results with class and bounding boxes and logs the result for each image for computation of evaluation metrics.
EvaluatePerformance.py	Constitutes helper methods that take ground truth data and results as input and compute different evaluation metrics for each category and summarize the results for each category to evaluate and return overall performance of each model in form of a pandas dataframe.
PipelineMain.py	Main module that executes YOLOv3 and Faster R-CNN models sequentially and prints the results on the console.

Table 4.1: Scripts for Object Detection task

For running Semantic Segmentation inference on the images, the module is written and executed on Jupyter notebook as the DeepLab Semantic Segmentation model requires TensorFlow version 1.x. The images are uploaded to the google drive to run the models for all the directory images. All the scripts with the pre-trained models used in the experiment are available at the github repository.

4.2 Object Detection – YOLOv3

4.2.1 Configuration

As stated earlier, YOLOv3 internally uses darknet architecture. Python package: OpenCV provides the implementation of darknet architecture for ease of use. Hence, making the configuration of the YOLOv3 model smooth. The weights for the pre-trained YOLOv3 model are available to download from this link.

There are multiple open resources available to provide a guided implementation for loading and executing the YOLOv3 model. SowmiyaNarayanan (49) provides succinct details about YOLO architecture, implementation to load pre-trained YOLOv3 model, and run the model for sample images. The experiment implementation takes inspiration from the article and reshapes the code to meet the task's goals.

The results from the YOLOv3 model are stored in the 'results/yolo_model/' directory. A pandas dataframe stores the summary of results for each inference to evaluate the performance across the categories.

4.2.2 Results

YOLO provides a robust approach for object detection among single shot detectors and is known to be at par with other models like Faster R-CNN.

Fig. 4.1 shows some sample instances where YOLOv3 performs as expected and is not confused by the suggested adversarial examples. In the top-left and bottom-right image, the model distinguishes between the billboard/mural and the person with confidences values of 99%. It also identifies the car decorated with 3D fruits (top-right) and the stop sign artistically altered (bottom-left) with confidence values of 87% and 95%, respectively.

Figure 4.1: YOLOv3 - successful examples.

(top-left) Man standing besides a billboard, (top-right) Car decorated with fruits, (bottom-left) Artistic recreation of Stop sign (bottom-right) Woman sitting in front of truck mural

Figure 4.2: YOLOv3 - Images acting as Adversarial Example

(top-left) Sculpture mistaken for a person on bicycle, (top-right) Creative bicycle parking confused as bicycle, (centre-left) Unidentified artistic stop sign, (centre-right) Mural with man on bicycle identified as real man on bicycle, (bottom-left) Undetected car textured as grass, (bottom-right) Mural containing cart and people identified as real cart and people

However, multiple road-related scenarios presented above reveal to confuse YOLOv3, sample instances presented in Fig. 4.2. The sample image on top-left shows a road scenario where an old cycle is revived by adding a human-like structure, and the model identifies it as a cyclist with a human with a confidence level of 96% and 95%, respectively. These static recreations along the road might confuse the autonomous vehicles to apply the emergency brake. A similar instance is an image on top-right; the creative cycle parking tends to puzzle the model and detects the parking as a cycle with a confidence value of 93%. On the contrary, images on centre-left and bottom-left present the cases that artistic creation around the stop sign and grass texture of the car deceives the model and fails to identify them. Additionally, the paintings on the walls of images on centre-right and bottom-right are identified as objects with a minimum confidence value of 96%.

These sample instances suggest that these real-world scenarios are highly likely to trick an autonomous vehicle into making a mistake.

The overall performance for YOLOv3 across all the categories is provided below in Table 4.2.

Category Name	**Accuracy**	**Precision**	**Recall**	**F1-score**	**TP**	**FP**	**FN**
On-road Scenario	0.61	0.88	0.67	0.76	467	61	232
Art-in-surrounding and Murals	0.24	0.25	0.78	0.38	76	223	22
Street Signs	0.5	0.8	0.58	0.67	51	13	37
Parking spaces	0.7	0.91	0.75	0.82	86	8	29
Vehicle Art and Textures	0.61	0.9	0.65	0.75	460	49	249
Total	**0.55**	**0.76**	**0.67**	**0.71**	**1140**	**354**	**569**

Table 4.2: Cumulative Results for YOLOv3 model

The high False Positives in the Art-in-surrounding and Murals indicates that multiple objects in the painting and murals are identified as actual objects in the surroundings. Whereas, for the outstanding categories, the high numbers in False Negatives show that there are multiple objects present in the surroundings that

the model cannot identify.

Moreover, the model's accuracy is low at 55% with Precision and Recall values as 76% and 67%, respectively. The stats in the table clearly show that the given images act as adversarial cases and significantly reduce the YOLOv3 model's performance.

4.3 Object Detection – Faster R-CNN

4.3.1 Configuration

There are multiple implementations of Faster R-CNN available open for use. We have used the TensorFlow version 2 implementation of the Faster R-CNN model from the TensorFlow 2 Detection Model Zoo. It provides various models trained on the COCO dataset to perform object detection.

The Faster R-CNN model used for testing the images is built on ResNet101 architecture with 1024x1024 resolution has an mAP of 37.6. The model is available for out-of-the-box use at link. TensorFlow also provides tutorials to use these models, as required, in the form of colab_tutorials. These tutorials are used as a reference to run inference on the images for this experiment with a threshold value of 70%.

The model's inference results are saved in the 'results/frcnn_model/' folder for further analysis. The results are also summarised and stored in a pandas dataframe to assist in the computation of evaluation metrics for the model.

4.3.2 Results

Faster R-CNN architecture provides a time-efficient region-based object detection technique. It is known to be slower than YOLOv3 but provides better results in

identifying smaller objects and details in the image.

Faster R-CNN identifies more details in the images than the YOLOv3 and hence, provides good results for many images in the dataset. Sample provided in Fig. 4.3. The images on top-left show the successful identification of a bicycle stand in front of the bus. The top-right and bottom-right images show the instances where YOLOv3 fails, and Faster R-CNN successfully identifies the stop sign and car covered with grass texture, respectively. The image on the bottom-left shows that the mural with cyclist does not confuse the model.

Figure 4.3: Sample images showing successful results for Faster R-CNN.
(top-left) Bus with cycle stand in front, (top-right) Modified stop sign on the road, (bottom-left) Mural with cyclist on the road, (bottom-right) Car covered with grass texture

Figure 4.4: Faster R-CNN - sample negative results.

(top-left) Photo of person in billboard identified as person, (top-right) Decorated car undetected by the model, (centre-left) Painting of motorcycle on wall identified as actual motorcycle, (centre-right) Custom-built car with texture identified as cake, (bottom-left) Shopping cart in the parking identified as bicycle, (bottom-right) Model confused by the creative figure built using discarded street signs

The sample results for the Faster R-CNN model in this experiment are shown in Fig. 4.4. For instance, images on top-left and centre-left, show that the Faster R-CNN model identified the objects from billboards and murals with high confidence values of 98% and 99%, respectively. The image on top-right successfully classifies all the cars on the road except the decorated car. The model misclassifies the image in centre-right and bottom-left high confidence levels. Additionally, the car on centre-right is identified as a cake with 97% confidence, whereas, the shopping cart is identified as a bicycle with a confidence value of 91%. The last instance (bottom-right) shows the discarded street sign for recreational art; the model identifies the stop signs with confidence levels of 97% and 90%. Moreover, it misinterprets the arrows as a parking meter with a confidence value of 90%.

Category Name	**Accuracy**	**Precision**	**Recall**	**F1-score**	**TP**	**FP**	**FN**
On-road Scenario	0.64	0.77	0.79	0.78	550	161	149
Art-in-surrounding and Murals	0.15	0.15	0.86	0.26	84	460	14
Street Signs	0.56	0.63	0.84	0.72	74	43	14
Parking spaces	0.78	0.82	0.94	0.88	108	24	7
Vehicle Art and Textures	0.59	0.72	0.76	0.74	542	212	167
Total	**0.52**	**0.60**	**0.79**	**0.68**	**1358**	**900**	**351**

Table 4.3: Cumulative Results for Faster R-CNN model

Even stats for Faster R-CNN in Table 4.3, show high values for misclassifications in each category. Similar to YOLOv3, Art-in-surrounding and Murals show high False Positives, implying that the model is confused by the painting and murals. Additionally, Faster R-CNN shows high False Positive for Vehicle Art and Texture as well. For the other categories, there are significant False Negatives—nonetheless, the False Negatives in lesser for Faster R-CNN than with YOLOv3.

The summary of results for the Faster R-CNN model gives multiple insights. The numbers for True Positives for Faster R-CNN is significantly higher than YOLOv3, leading to comparatively lesser False Negatives. However, Faster R-

CNN has massive False Positives, which attributes for a reduction in the model's overall performance. Hence, the model's accuracy is very low with 52%, precision is 60%, and recall is 79%. On the whole, the scenarios adversely affect both Faster R-CNN and YOLOv3 models.

4.4 Semantic Segmentation – DeepLabv3

4.4.1 Configuration

The DeepLab series by Google is available at GitHub. The model is pre-trained on the COCO dataset and uses colour map from the PASCAL VOC dataset with 21 output labels (background + 20 PASCAL VOC labels).

The DeepLabv3 model used for this project operates on MobileNetv2 architecture with depth-multiplier as 0.5 has IOU of 77.33% and 75.32% on the validation set for output strides 8 and 16, respectively. The link above provides different models trained on PASCAL VOC, Cityscapes dataset, etc. ready to use. It also provides a demonstration to run inferences using the model in the form of colab notebook. The provided framework is used for the setup of the model for this research as well.

As the given model internally uses the TensorFlow version 1 implementation, the inference is carried out in Google colab to achieve the required configuration for the model and project uses google drive for storing the input images and the output inferences.

4.4.2 Results

As semantic segmentation is an extension to the proposed project, it requires pixel-level details for evaluation, and is employed to run inference only for 19

images; the model is evaluated manually for each of the instances. The output image includes the input image, segmentation map, and overlap of the segmentation map with the input image.

The output of segmentation for some images show expected results, i.e., the model is potentially not affected by the presence of the illusions around them. Sample instance on the top in Fig. 4.5 shows an illusion of the road in the form of ice blocks and a man fishing on them. The image segmentation only identifies the people in the image and successfully ignores the art that creates the illusion.

Figure 4.5: DeepLabv3 Semantic Segmentation sample instances.
(top) Illusion of a broken road and a man fishing between the road , (bottom) People standing besides an illusion of road tearing through the wall and cyclist going in.

Another instance provided on the bottom in Fig. 4.5 depicts a bicycle painting on the wall creating an illusion of a wall tearing apart and the cyclist riding into it with people besides it. The image results show that the model identifies the cyclist and the bicycle with the people standing beside them. This example and some other instances imply that the images can confuse the semantic segmentation model.

On analysing the results for all 19 images, we can observe that the ignorance of the illusion on the top in Fig. 4.5 and similar images can be attributed to lesser output classes available in the PASCAL VOC dataset. Most images' semantic segmentation results imply that the model identifies depths for the illusion or paintings created. However, due to the limited availability of the image data and lack of a comprehensive model specifically trained for road or transportation scenarios, we cannot provide an overall generic outcome for the semantic segmentation task.

Chapter 5

Conclusion and Future Work

This chapter summarizes the results of the research and suggests some proposals for future work.

5.1 Conclusion

From the models' inference results, we can conclude the following for each of the studied categories:

- **Art-in-surrounding and Murals**: The paintings and Murals from this category confuse the models the most. The YOLOv3 and Faster R-CNN models show the worst performance, with a high number of False Positives stating that the models identify the paintings' components as real objects.
- **Street Signs**: The art creation around the street sign and existence of road signs in regional variations degrades the performance for both the models. Though the models are trained for limited class labels, the limited number of examples provide some insightful scenarios; to encourage further identification and analysis of similar scenarios.

- **Vehicle Art and Textures**: The fancy vehicles presented in this category show contrasting results for Faster R-CNN and YOLOv3 models. While the YOLOv3 model cannot identify the objects in the image (high False Negatives), the Faster R-CNN identifies the objects that are not present (high False Positives). Hence, showing that the YOLOv3 does not capture enough information and Faster R-CNN captures too much information to confuse it.

- **On-road scenarios**: The un-seen road scenarios presented in this category show comparatively better performance for the models. Both models have a high number of False Negatives, indicating that the models are unable to identify the objects in these new scenarios.

- **Parking spaces**: Parking spaces, with the least use-cases, shows the best performance for the models among the given categories. The models, possibly, are not misled due to limited class labels from the COCO dataset or the limited scenarios presented in the research. However, it puts forward a few interesting adversarial scenarios for the self-driving applications.

Analysing the aggregate results for both the models shows that the Faster R-CNN model captures more details than the YOLOv3 model. This helps the Faster R-CNN model make better inferences; however, it also makes the Faster R-CNN model more vulnerable to the adversarial attacks.

The research recognizes multiple natural scenarios that act as adversarial examples for the current state-of-the-art object detection models, namely, YOLOv3 and Faster R-CNN.

5.2 Future Work

The current research though successful, has a huge scope for further extension and analysis. Some limitations of the project are:

- **Limited class labels**: As stated earlier, the COCO dataset has 80 only commonly available labels. Multiple instances in the research identify the need to use specialised dataset for better analysis.
- **Restricted Image Format**: As the pre-trained models identify only JPEG format, all the images in the dataset use the same format. Inclusion of other image formats such as TIFF or PNG can further add new scenarios, improving the dataset's quality.

These limitations, however, can be viewed as an opportunity for further improvement of the presented dataset. Some suggestion for future work on this research is:

- **Advanced Scenarios**: The Advanced Scenarios presented in the dataset can be assessed further using a more comprehensive model trained specifically for road scenarios.
- **Extensive Evaluation**: Due to limited time, this research only evaluates the results of object detection based on the number of objects present in the image. The research can further be extended to evaluate the bounding boxes of the images using Intersection-over-Union (IoU) approaches.
- **Identification of additional scenarios**: Few articles on the internet depict alternate approaches for identifying adversarial examples for autonomous vehicles. For example, Mufson (50) presents art creation by Filip Piekniewski containing scenarios that can confuse the autonomous vehicles. Similar articles can be used to refine the collected dataset further.

References

[1] J. Janai, F. Güney, A. Behl, and A. Geiger, "Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art," *arXiv e-prints*, p. arXiv:1704.05519, Apr. 2017. iii, 1, 9, 11, 12, 13, 14, 15

[2] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song, "Natural adversarial examples," *arXiv preprint arXiv:1907.07174*, 2019. iii, viii, 2, 16, 17, 18

[3] D. Feng, C. Haase-Schütz, L. Rosenbaum, H. Hertlein, C. Glaeser, F. Timm, W. Wiesbeck, and K. Dietmayer, "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges," *IEEE Transactions on Intelligent Transportation Systems*, 2020. iii

[4] Y. S, "Build up a neural network with python," May 2020. [Online]. Available: https://towardsdatascience.com/build-up-a-neural-network-with-python-7faea4561b31 viii, 7

[5] J. Padarian, B. Minasny, and A. McBratney, "Using deep learning to predict soil properties from regional spectral data," *Geoderma Regional*, vol. 16, p. e00198, 2019. viii, 8

[6] "Common objects in context." [Online]. Available: https://cocodataset.org/#home viii, 15, 16

[7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013. viii, 2, 16, 17

[8] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 427–436. viii, 2, 16, 17, 18

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. viii, 19, 20

[10] A. Kathuria, "What's new in yolo v3?" Apr 2018. [Online]. Available: https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b viii, 21

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587. viii, 10, 22

[12] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448. viii, 10, 22, 23

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99. viii, 7, 10, 23

[14] R. Gandhi, "R-cnn, fast r-cnn, faster r-cnn, yolo - object detection algorithms," Jul 2018. [Online]. Available: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e viii, 22, 23

[15] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017. viii, 24, 25

[16] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017. viii, 25

[17] S. A. Papert, Jul 1966. [Online]. Available: https://dspace.mit.edu/handle/1721.1/6125 1

[18] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010. 1, 9

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015. 2

[20] M. Hargrave, "How deep learning can help prevent financial fraud," Jul 2020. [Online]. Available: https://www.investopedia.com/terms/d/deep-learning.asp 7

[21] S. Kostadinov, "Understanding backpropagation algorithm," Aug 2019. [Online]. Available: https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd 7

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788. 7, 10, 19

[23] H. Pokharna, "The best explanation of convolutional neural networks on the internet!" Jul 2016. [Online]. Available: https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8#:~:text=CNNs,likeneuralnetworks,are,andrespondswithanoutput. 8

[24] "Convolutional neural network," Sep 2020. [Online]. Available: https://en.wikipedia.org/wiki/Convolutional_neural_network 8

[25] "Computer vision," Sep 2020. [Online]. Available: https://en.wikipedia.org/wiki/Computer_vision 9

[26] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 886–893. 10

[27] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 12, pp. 2179–2195, 2008. 10

[28] "Getting started with semantic segmentation using deep learning." [Online]. Available: https://www.mathworks.com/help/vision/ug/getting-started-with-semantic-segmentation-using-deep-learning.html 11

[29] "Visual object classes challenge 2012 (voc2012)." [Online]. Available: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/#devkit 11

[30] "Image thresholding." [Online]. Available: https://www.mathworks.com/discovery/image-thresholding.html 11

[31] B. Sahu, "The evolution of deeplab for semantic segmentation," Aug 2019. [Online]. Available: https://towardsdatascience.com/

the-evolution-of-deeplab-for-semantic-segmentation-95082b025571 11, 12, 24

[32] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020. 13

[33] M. Minsky, "Steps toward artificial intelligence," *Proceedings of the IRE*, vol. 49, no. 1, pp. 8–30, 1961. 14

[34] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013. 14

[35] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223. 14

[36] "The cityscapes dataset." [Online]. Available: https://www.cityscapes-dataset.com/ 14, 15

[37] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The apolloscape dataset for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 954–960. 15

[38] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755. 15

[39] C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le, "Adversarial examples improve image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 819–828. 19

[40] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017. 19

[41] J. Redmon. [Online]. Available: https://pjreddie.com/darknet/yolo/ 19

[42] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018. 20

[43] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014. 24

[44] "Arcgis api for python." [Online]. Available: https://developers.arcgis.com/python/guide/how-deeplabv3-works/#:~:text=DeepLabV3ModelArchitecture&text=Ontopofextractedfeatures,segmentedmaskfortheimage. 25

[45] S. Koço and C. Capponi, "On multi-class classification through the minimization of the confusion matrix norm," in *Asian Conference on Machine Learning*, 2013, pp. 277–292. 26

[46] R. Khandelwal, "Evaluating performance of an object detection model," Jan 2020. [Online]. Available: https://towardsdatascience.com/evaluating-performance-of-an-object-detection-model-137a349c517b 27

[47] "Murals in northern ireland," Apr 2020. [Online]. Available: https://en.wikipedia.org/wiki/Murals_in_Northern_Ireland 31

[48] "2020 houston art car parade." [Online]. Available: https://www.thehoustonartcarparade.com/ 34

[49] G. SowmiyaNarayanan, "Object detection using yolov3," Jun 2020. [Online]. Available: https://towardsdatascience.com/object-detection-using-yolov3-9112006d1c73 51

[50] [Online]. Available: https://www.vice.com/en_au/article/yp7dmw/engineer-painting-surreal-self-driving-car-scenarios 64

Appendix A

List of Image Sources

All the images collected in this projects are publicly available. For mining images from internet, Google images filter 'Labeled for reuse' has provided great help.

The source of all the images in the dataset, in no particular order is:

- Pikist
- Public Domain Pictures
- Pixabay
- Geograph
- Wikimedia Commons
- Pexels
- PxHere
- PickPik
- PxFuel
- NeedPix

Appendix B

Python Scripts

The code written as part of this research has been uploaded to github link: UniversalAdversarialChallenges-AutonomousVehicles

The repository is divided in 4 folders and an Excel file:

- **/data**: The data folder is the heart of the project as it consists of all the images collected for the research.
- **/models**: This folder contains pre-trained models and supporting files for running the inference on state-of-the-art models
- **/results**: The inference results for all the images are stored in this folder
- **/scripts**: This folder consists of all the python files used to run inference for object detection and semantic segmentation tasks.
- **GroundTruth.xlsx**: This file contains the ground truth details for all the images in data folder to evaluate the inference results