HUMAN-CENTRIC SMART CITIES: A DIGITAL TWIN-ORIENTED DESIGN OF INTERACTIVE AUTONOMOUS VEHICLES

A Thesis

by

OSCAR G. DE LEON VAZQUEZ

Submitted in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Major Subject: Mechanical Engineering

The University of Texas Rio Grande Valley

December 2023

HUMAN-CENTRIC SMART CITIES: A DIGITAL TWIN-ORIENTED DESIGN OF INTERACTIVE AUTONOMOUS VEHICLES

A Thesis
by
OSCAR G. DE LEON VAZQUEZ

COMMITTEE MEMBERS

Dr. Fatemeh Nazari
Co-chair of Committee

Dr. Constantine Tarawneh
Co-Chair of Committee

Dr. Mohamadhossein Noruzoliaee
Committee Member

Dr. Horacio Vasquez
Committee Member

December 2023

ABSTRACT

De Leon Vazquez, Oscar G., Human-Centric Smart Cities: A Digital Twin-Oriented Design of Interactive Autonomous Vehicles. Master of Science in Engineering (MSE), December, 2023, 51 pp., 7 tables, 7 figures, references, 29 titles.

Autonomous vehicle (AV) technology is introduced as a solution to improve transportation safety by eliminating traffic accidents caused by human error, which is the leading cause of 90% of accidents. One key feature of AVs is sensing and perceiving their surrounding environment through processing observations collected from the environment. The perception system is essential for an AV to make informed decisions and safely navigate the environment. This study presents an image semantic segmentation algorithm developed in the area of computer vision to improve AV perception. The U-Net-based algorithm is trained and validated using a synthetically generated dataset in a simulation environment, namely, CAR Learning to Act (CARLA). The results indicate an improved accuracy of up to 98% compared to the existing methods. The performance of the proposed model is further analyzed using various evaluation metrics.

DEDICATION

This thesis is dedicated to those who have profoundly influenced my life:

To my father, Gilberto De Leon, and my mother, Irma Vazquez, whose unwavering love, and guidance have shaped me into the person I am today. I extend my gratitude to the faculty of the College of Engineering for their invaluable insights and support. Lastly, my heartfelt thanks go to my friends and colleagues who stood by me throughout my academic journey.

ACKNOWLEDGEMENTS

This work was made possible through the support of the National Science Foundation CREST Center for Multidisciplinary Research Excellence in Cyber-Physical Infrastructure Systems (MECIS), funded by NSF Award No. 2112650. I would also like to express my appreciation for the support provided by Drs. Nazari, Noruzoliaee, and Tarawneh. Additionally, I extend my gratitude to my coworker and friend Leonel Ramirez for his valuable contributions and support.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

Human errors are the primary cause of accidents on the road, underscoring the urgency of developing autonomous technologies to mitigate such risks (Singh, 2015). These technologies will improve road safety, increase productivity, and transform the future of transportation. Autonomous technologies, driven by artificial intelligence and advanced sensing capabilities, offer a promising solution to these challenges (Kuutti et al., 2021). By lowering human error, autonomous technologies will increase road safety and make all users' experience on the roads safer. Autonomous vehicles (AVs) are trained to perform at a level of accuracy and attentiveness that is often difficult for humans to match when it comes to avoiding collisions with other cars, navigating hazardous weather, or reacting to unforeseen obstacles.

Furthermore, integrating AVs into our transportation ecosystem promises to increase traffic efficiency (Martinez-Diaz & Soriguera, 2018). One of the most essential advantages of AVs is their potential to reduce traffic, especially in many urban areas (Talebpour & Mahmassani, 2016). Researchers aim to create AV technologies that can communicate with each other to maintain a consistent distance between vehicles. This coordinated movement optimizes traffic flow by minimizing needless lane changes and abrupt stops, lowering aerodynamic drag, and increasing fuel efficiency.

All these are just some of the few benefits AVs will have in our ecosystem. Although the future is promising, researchers and engineers need help reaching level 5 of automation, which is the maximum level without human intervention (Yaqoob et al., 2018). AV technologies present challenges such as safety concerns, infrastructure adaptation, cost and accessibility, and ethical considerations. One of the biggest challenges for engineers is the variability of real-world scenarios. The real world is highly dynamic and unpredictable. Researchers are trying to duplicate human perception in autonomous technologies. However, human perception is characterized by the ability to adapt to complex and unpredictable situations. On the other hand, AVs are limited by the amount of data. They rely on a variety of sensors to handle real-world scenarios. These sensors, such as cameras, LiDAR, and radar, are required to perceive the environment.

This thesis aims to contribute to this transformative era by addressing specific challenges related to perception and semantic segmentation in autonomous vehicles, thereby playing a pivotal role in realizing a safer, more efficient, and sustainable transportation future.

Statement of the Problem

AV perception systems exhibit accuracy and real-time processing limitations, which are crucial for safety in urban settings. The challenge lies in developing an image semantic segmentation algorithm to process and interpret complex visual data in real-time accurately. Most traffic accidents can be traced back to erroneous human perceptions and decisions. As a result, the reliability of AV's perception systems is a critical safety and technical concern.

Purpose of the Study

The purpose of this study is to design, develop, and validate an image semantic segmentation algorithm that significantly enhances the perception capabilities of AVs in simulated smart city environments. This research is driven by the imperative to improve the safety features of AVs, thereby reducing the risk of accidents attributed to the limitations of current perception systems. Through the use of a U-Net-based algorithm and a synthetic dataset generated within the CARLA simulation environment, the ambition is to achieve a remarkable enhancement in the perception capabilities of AVs. By conducting the training and testing of the model within the simulator, this study avoids the risks and ethical dilemmas that come with real-world testing.

The second purpose of this study is to contribute to the field of smart city development. By improving the perception systems of AVs. The study aims to facilitate the integration of AVs into urban environments, where they can function efficiently and autonomously.

Significance of the Study

The significance of the study lies in the technological advancement and social benefit of focusing on the development of image semantic segmentation algorithms for AVs. The potential to reduce traffic accidents underscores the profound impact this study could have on public safety and the quality of urban life. Additionally, this study explores the territory of applying sophisticated computer vision techniques in a simulated environment that closely mirrors real-world urban settings. The success of this algorithm in a simulated environment could serve as a benchmark for future AV technologies, offering a reliable blueprint for safe, effective, and ethical AV development. Moreover, the study has the potential to substantially contribute to the

smart city paradigm, which emphasizes the importance of fully integrating technology and urban infrastructure. This research aligns with the objectives of smart cities to optimize resource use, reduce congestion, minimize environmental impact, and improve overall urban livability. The contributions of this thesis are:

- The thesis involves implementing a U-Net semantic segmentation model, demonstrating high accuracy in segmenting urban scene elements like roads, sidewalks, vehicles, and pedestrians. The model's training, validation, and test performances highlighted its robustness and potential for real-world application.
- The thesis employs synthetic data from the CARLA simulator and utilizes the MICC-SRI Semantic Road Inpainting Dataset for its comprehensive collection of road scenes. This approach ensures quality and diversity in the input data, which is crucial for the accuracy and reliability of semantic segmentation models.
- The study extensively reviews hyperparameters used in past studies, focusing on tuning learning rate, epochs, and batch size. This process includes random and grid search methods to determine the best-performing hyperparameters.

Motivation for the Thesis

The motivation of this thesis is to solve the challenges of autonomous technologies, particularly the limitations of existing AV perception systems. This thesis is driven by the ambition to push the boundaries of computer vision and machine learning in the domain of AVs and to do so within the ethical constraints of simulated environments that reflect the complexity of urban life.

Organization of the Thesis

This thesis is organized into five chapters, each focusing on a different aspect of integrating autonomous vehicles in smart cities. Chapter I introduces the topic and sets the context for the research. Chapter II reviews relevant literature, covering critical computer vision and semantic segmentation developments. Chapter III details the methodology, explaining how the semantic segmentation model was developed and trained. Chapter IV presents the results, analyzing the model's performance and effectiveness. Finally, Chapter V concludes the thesis, summarizing the findings and suggesting directions for future research. This structure guides the reader through the study's clear and logical progression.

CHAPTER II

LITERATURE REVIEW

Image semantic segmentation is a fundamental task in computer vision, with applications ranging from autonomous driving to medical image analysis. This section reviews the existing literature by highlighting significant approaches, advancements, and challenges in three parts. The first part reviews studies on deep learning methods in the area of computer vision for object detection and semantic segmentation. Turning to the context of the study, which is focused on autonomous driving systems, the second part reviews the relevant studies using the CARLA simulation. The last part discusses the present study in comparison with the research studies.

Computer Vision for Object Detection and Semantic Segmentation

Computer vision is a subset of artificial intelligence that uses computers to understand visual information from the world, using a similar approach to human visual systems. Among the existing four sub-fields, this study focuses only on object detection and semantic segmentation (the areas of image classification and instance segmentation are out of the scope of this study). Accordingly, this section delves into various algorithms pertinent to visual perception tasks, exploring their functionalities and applications in the context of this specialized area. It provides a comprehensive overview of current methodologies, examining how they contribute to advancements in object detection and semantic segmentation. This exploration not only

highlights the strengths and limitations of existing approaches but also identifies potential areas for future innovation and research within the realm of computer vision.

Object Detection

Object detection is a computer vision technique focusing on detecting and locating objects within an image or video. It involves teaching a computer to identify and depict bounding boxes around particular visual data objects of interest. The first breakthrough is named AlexNet, which is a deep learning neural network introduced by Krizhevsky et al. (2012) at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012. This CNN model makes significant progress in the deep learning field by demonstrating that deep CNNs can outperform the conventional computer vision methods on image classification. This study wins the ILSVRC competition with a test accuracy of 84.6%, meanwhile, their closest competitor achieved 73.8% accuracy.

Two years later, GoogleNet by Szegedy et al. (2015) wins the same competition with a test accuracy of 93.3%. It comprises 22 layers and introduces a new term called inception module. This module helps to capture features in different levels of abstraction. By doing this, the neural network can capture information while effectively allocating computational resources.

Previous studies concentrated on identifying a specific category of provided objects. However, more recent research utilizes deep learning methods to advance object detection. For instance, R-CNN by Ross Girshick et al. (2014), which stands for region-based Convolutional Neural Network, combines the power of deep learning (CNNs) with conventional computer vision techniques (region proposals and classifiers), which at the time represented a significant advancement in object detection. Using the PASCAL VOC 2012 dataset, R-CNN outperforms

earlier state-of-the-art techniques regarding object detection accuracy, as indicated by mean average precision (mAP). R-CNN generates a mAP of 53.3%, significantly improving on the previous best result. Unfortunately, because this model requires multiple steps, it is computationally expensive and unsuitable for real-time applications.

Object detection with CNNs becomes more feasible for real-world applications with the introduction of Fast R-CNN and Faster R-CNN, two subsequent advancements in this field that addressed some of these limitations by combining the region proposal generation and feature extraction into a more efficient end-to-end model. Fast R-CNN (Girshick, 2015) improves the detection efficiency of R-CNN by using a Region of Interest (RoI) pooling layer to extract features from the proposed regions, resulting in faster processing. A better version is Faster R-CNN (Ren et al., 2015), which combines the use of Regional Proposal Network (RPN) and Fast-RCNN into an end-to-end network. This model achieves state-of-the-art performance for multiple datasets. Faster-RCNN achieves a mAP that exceeds 70% on the PASCAL VOC dataset on the VOC 2007 and VOC 2012 test sets.

Subsequently, Mask R-CNN (He et al., 2017) is introduced as a continuation of Faster R-CNN, merging object detection with pixel-level instance segmentation. In the context of MS-COCO in 2017, Mask R-CNN demonstrates the highest levels of detection accuracy by employing ResNet101-FPN as its backbone network architecture.

While highly accurate, two-stage R-CNN architectures are inefficient in real-time applications due to their longer processing time. This circumstance leads to the preference for one-stage architectures. In order to avoid the need for a second region proposal stage, one-stage architectures seek to detect objects in a single network pass. YOLO (Redmon et al., 2016) is one of the pioneers in single-stage detection. YOLO formulates the object detection task as a

regression problem rather than as a classifier problem. The architecture of YOLO is based on a CNN that consists of two fully connected layers after 24 convolutional layers. In the output, YOLO forecasts bounding box coordinates for every grid cell based on the location of the cell. For every bounding box, an object score is predicted to show whether an object is present in that grid cell. It assists in removing blank or meaningless boxes. YOLO achieves a mAP of 63% on the PASCAL VOC 2007 dataset. In addition, YOLO can reach 45 frames per second in real-time, making it fast enough to compete with R-CNN architectures. However, one of the significant limitations of YOLO is that it only predicts two boxes and one class and has a lower accuracy than two-stage approaches. Another example of a one-stage object detection method is the single-shot MultiBox detector (SSD) (Leibe et al., 2016). It outperforms the two-stage method because it eliminates the need for a separate region proposal network. It has a competitive efficiency compared to YOLO and achieves 76.9% mAP on the PASCAL VOC 2007 dataset. Although this method is fast and accurate, newer versions of YOLO outperform it because they have demonstrated faster processing times and higher accuracy.

Later iterations of YOLO are built on top of it, with the goal of improving its shortcomings without sacrificing its effectiveness. For instance, YOLOv2 (Redmon & Farhadi, 2017) outperforms the original by implementing a classification method. YOLOv2 is optimized for classification and detection tasks, leading to a versatile model capable of handling up to 9000 elements. The result of YOLOv2 shows a 76.8% mAP on the PASCAL VOC 2007 dataset, representing a significant advancement over its predecessor. The integration of Darknet-19 into YOLOv2 exemplifies the model's balance between accuracy and speed, affirming its suitability for real-time object detection.

Another better version of YOLO is YOLOv3 (Redmon & Farhadi, 2018). This newer version better represents small objects, although it performs less on medium and larger objects. Instead of using Darknet-19, this version of YOLO utilizes darknet-53, which is a more complex and deeper convolutional neural network. This version introduces a multi-label classification, which was a significant improvement in comparison with older versions. Additionally, YOLOv3 improves efficiency by adding a score for each bounding box. While there are plenty of exciting advancements in object detection, such as the latest YOLO versions, this study does not dive into all of them. The focus of this study is on semantic segmentation, which is reviewed in the next part of this section.

Deep Learning-based Semantic Segmentation

In the particular case of semantic segmentation models, the process can be slightly different. As mentioned before, semantic segmentation is based on the association of labels with every pixel in the image. The input image is preprocessed separately, and it is divided into pixels. In the field of semantic segmentation, fully convolutional layers (FCNs) represent a prevalent architectural choice. FCNs are used by replacing any fully connected layer with a convolutional layer. By doing this, the model requires fewer parameters, which makes the networks more accesible to train (Long et al., 2015). FCN has flexibility but needs more precision and accuracy due to the high-level maps. This constraint causes the final maps to lack the detail of objects.

In addition to the FCN architecture, researchers develop other variations to change a network intended for classification into one appropriate for segmentation. One of these variations is the encoder-decoder network, composed of three main steps. The encoder, the initial stage, is responsible for taking a network and classifying it by deleting any fully connected layers.

Feature map representations in low resolution are produced as a result. The second stage is the bottleneck, which is responsible for transitioning from the encoder to the decoder. This layer compresses the information to preserve the relevant information needed to reconstruct the input data. The last stage is the decoder layer, which is responsible for upsampling the feature maps to correspond with the spatial dimensions of the image to reconstruct the output image.

Noh et al. (2015) introduce a novel semantic segmentation algorithm called DeconvNet, comprising an encoder and decoder. Based on VGG16 architecture, the encoder extracts feature vectors, while the decoder utilizes deconvolution to generate pixel-wise probability maps, maintaining class information and spatial details.

DeepLabv1 (Chen et al., 2015) utilizes dilated convolutions, which is a kind of convolution operation that expands the receptive field of CNNs without adding more parameters, to precisely control feature map resolutions in Deep CNNs, while DeepLabv3 (Chen et al., 2018) introduces spatial pyramid pooling, enabling object segmentation at multiple scales by employing filters with diverse sampling rates and effective fields-of-views.

Ronneberger et al. (2015) introduces U-Net, an extended version of FCN designed for biological microscopy applications, featuring encoding and decoding components. U-Net utilizes skip connections between the encoder and decoder to maintain pattern information, resulting in a relatively small number of weights and enabling fine-grained segmentation. This model is the one used for this study. A more detailed explanation of the model's structure will appear in Chapter 3.

Autonomous Driving Research

Through the development of object detection and semantic segmentation models, computer vision plays a critical role in the advancement of autonomous vehicles. These models equip vehicles with the capacity to perceive and gain insight into their surroundings, empowering them to make well-informed decisions and safely navigate intricate environments. This section provides insight into different object detection and semantic segmentation models in the autonomous driving field.

In the field of autonomous driving research, Niranjan et al. (2021) propose an object detection model for the classification of 5 different classes (vehicles, bikes and motorcycles, traffic lights, and traffic signs). The study emphasizes the efficiency of using only cameras compared to other sensors like LiDAR and RADAR. This research paper uses the SSD algorithm, which has an efficient CNN structure and obtains an accuracy of 82.81%. Furthermore, the paper highlights using open-source simulators like CARLA (Dosovitskiy et al., 2017). This simulator for autonomous driving research provides the user with a variety of sensors that can be used for the training and testing of deep learning models. The author of this paper uses CARLA to generate the dataset first by using five different maps. These same images were labeled for training purposes. The advantages of using a simulator like CARLA are that it offers an open-source API, robustness, and real-world environment modeling.

The same authors propose a paper with two object detection algorithms, SSD and Faster RCNN, for autonomous driving applications (Niranjan et al., 2021). Results show that SSD has 89% mean average precision and Faster RCNN has 95%. However, SSD has a faster computational speed of 30 ms/image than the 109 ms/image of the Faster RCNN. Moreover, the models that are trained on the CARLA simulator are also tested to detect real-world vehicles

with an accuracy of 60-70%. This flexibility demonstrates that trained models in driving simulators can be applied to real-world applications.

Additionally, a recent paper by Gao et al. (2021) presents an innovative approach to object detection in the context of autonomous driving. The authors argue that relying solely on real-world automatic driving vehicles for research is costly, less feasible, and challenging to test. Instead, they propose an efficient and cost-effective object detection method based on the CARLA simulator.

The Present Study

This section synthesizes the findings from Parts 2.1 and 2.2. It compares the current study with the context of the existing literature on semantic segmentation for object detection, mainly focusing on applications in autonomous driving.

A critical development in the application of CNNs for real-time semantic segmentation is presented in a research study by Paszke et al. (2016). This work introduces a novel deep network architecture named ENet (efficient neural network) that finds an equilibrium between accuracy and speed. This study was tested on the Cityscapes dataset, which consists of 5000 finned-annotated images. The dataset consisted of 19 classes, and the authors use the intersection over union (IoU) to evaluate their results. The results are compared to the SegNet (Badrinarayanan et al., 2016), where ENet shows poor performance, although it was faster than SegNet.

Our study exhibits a marked improvement in segmentation accuracy in static images without compromising other metrics. Our model significantly improves IoU scores, achieving over 0.95 for several classes and outperforming Enet’s reported scores. In addition, Enet's study only reports metrics IoU, which shows a big disadvantage in their model.

Another relevant study is Speeding Up Semantic Segmentation for Autonomous Driving by Treml et al. (2016). This study proposes a semantic segmentation model that maintains high accuracy. The proposed architecture includes the use of ELU (exponential linear unit) activation functions, a squeezeNet-like encoder, and a decoder with SharpMask. This model is tested on the Cityscapes dataset, a similar benchmark dataset from the previous study. The model was compared to ENet and SegNet by using IoU metrics. The results showed that their model outperformed both ENet and SegNet models. Although the new model achieved a higher accuracy than ENet, it is still falling behind the proposed model of this study. Our model focuses on static images and significantly improves accuracy using an encoder-decoder architecture that achieves high IoU scores.

Finally, this section revises a third study focusing on a practical deep fully convolutional neural network architecture for semantic pixel-wise segmentation by SegNet. The architecture of the model is composed of an encoder-decoder network followed by a classification layer. 13 convolutional layers are what the encoder network is made up of. These layers correspond to the VGG16 network (Simonyan et al., 2015), and their primary use is for object classification. After this, the encoder oversees upsampling input feature maps and provides reconstruction to high-resolution details. This is done by gradually increasing the image to the original size but decreasing spatial details to produce an output.

SegNet presents higher accuracy than ENet but lower performance than Treml's proposed study. Compared with our current model, SegNet's results show much lower IoU scores than ours. For instance, in some classes like fences, motorcycles, and traffic lights, the IoU scores are relatively low compared to our model, which scores higher with values of 35.7, 38.8, and 52, respectively.

This section compares our study with current semantic segmentation models. Our model excels in segmentation accuracy and achieves high IoU scores. ENet, while efficient, lacks performance compared to other models. SegNet stands out for achieving better in some classes than ENet but still shows low scores Finally, the work done by Treml improved only the ENet results on all but five classes but still struggled with traffic lights and signals, which are critical elements of autonomous driving. A major disadvantage for all three studies reviewed in this section is that these studies evaluated their models in IoU scores only. Our model is evaluated in many more metrics than just IoU scores. Table *1* illustrates the name of the research papers with their respective models and types of datasets. These studies are discussed in this chapter, along with the benefits and shortcomings of each paper. In conclusion, our model outperforms these studies with a high difference in accuracy values for each class.

Table 1: List of research papers using deep learning for computer vision applications

ID #	Name	Model	Application	Dataset	Data Type	Number of Data
RP1	Deep Learning based Object Detection Model for Autonomous Driving Research using CARLA Simulator.	SSD MobileNet (Single Shot Multibox Detector)	Object Classification in CARLA	Carla	RGB	1028
RP2	Performance Analysis of SSD and Faster RCNN Multi-class Object Detection Model for Autonomous Driving Vehicle Research Using CARLA Simulator.	SSD and Faster RCNN	Object Detection in CARLA	Carla	RGB	1000
RP3	An object detection research method based on CARLA simulation	YOLOv4; CenterNet; and Faster RCNN	Object Detection	Carla	RGB	3000
RP4	Stereo R-CNN based 3D Object Detection for Autonomous Driving.	Stereo R-CNN	a 3D object detection	KITTI	RGB	7841
RP5	2D object detection and semantic segmentation in the Carla simulator	YOLOv4 for RGB and ESPnetv2 for semantic segmentation	2D Object Detection	Carla	RGB	400
RP6	Semantic Segmentation with Carla Simulator	UNet	Semantic Segmentation	Carla	Semantical labeled synthetic	1000
RP7	Semantic Concept Testing in Autonomous Driving by Extraction of Object-Level Annotations from CARLA	DeepLabv3+	Semantic Segmentation	Carla	RGB, semantic segmentation	10099
RP8	Instance Segmentation in CARLA: Methodology and Analysis for Pedestrian-oriented Synthetic Data Generation in Crowded Scenes	----	Instance Segmentation	Carla	RGB, semantic, depth map, segmentation map	6532
RP9	Semantic Road Segmentation using Deep Learning	PSPNet; FCN; SegNet	Semantic Road	CityScapes	RGB	5000, 20000

CHAPTER III

METHODOLOGY

This chapter discusses the methodology used in this thesis to address the challenge of semantic segmentation in the context of autonomous driving. The chapter details the systematic approach taken to choose data, select the model, and the training procedure.

Data Collection and Preprocessing

The accuracy and reliability of semantic segmentation models for autonomous driving heavily depend on the input data's quality and diversity. This study employs the use of synthetic data generated from the CARLA simulator.

Dataset Selection

This study utilizes the MICC-SRI Semantic Road Inpainting Dataset (Berlincioni et al., 2019) due to its comprehensive collection of road scenes, which are crucial for training semantic segmentation models. It contains 11,913 frames, and all are available for RGB images and semantic segmentation. The dataset contains dynamic and static objects. The dynamic objects are obtained by driving the autopilot function within CARLA with pedestrians and vehicles. Additionally, the dataset includes various weather conditions and lighting scenarios, enhancing the model's ability to perform under different environmental settings.

Furthermore, the dataset has been pre-annotated with high precision, ensuring an accurate representation of road elements such as lanes, signs, and traffic lights.

Data Annotation

The MICC-SRI dataset provides pre-annotated data, which includes pixel-level labels necessary for training semantic segmentation models. In order to check the quality and consistency of these annotations, a sample of images is reviewed to verify the consistency of the labeling. This set was chosen randomly and Figure 1 displays the images for RGB, segmentation, and image overlay.

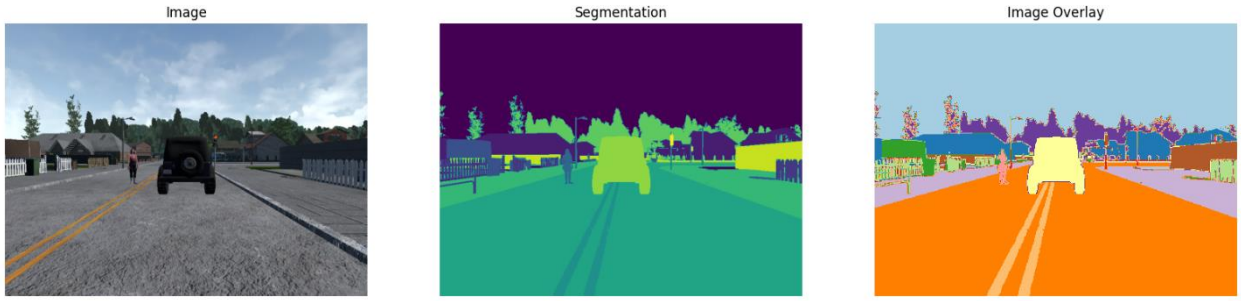

Figure 1: Preview random masked and unmasked images

The dataset obtained from the CARLA simulator has tags that are represented in Table 2. The set of classes is divided into dynamic and static objects. Dynamic entities are pedestrians and vehicles (including trucks, buses, cars, bikes, and motorbikes). On the other hand, static objects are buildings, fences, road lines, poles, sidewalks, vegetation, walls, traffic signals, and unlabeled objects. Chapter 4 introduces the evaluation of each class in a set of metrics.

Table 2: Class labels for urban elements

Value	Tag
0	Unlabeled
1	Buildings
2	Fences
3	Other
4	Pedestrians
5	Poles
6	Road Lines
7	Roads
8	Sidewalks
9	Vegetation
10	Vehicles
11	Walls
12	Traffic Signs

Data Preprocessing

A good performance semantic segmentation model depends on the quality of the dataset. The data undergoes a preprocessing phase to have a high-quality dataset. This subsection explains the following preprocessing steps.

Image Normalization. The images are normalized, which helps in stabilizing the learning process and achieving faster convergence during model training. The normalization process scales the pixel values to a range of 0 to 1.

Resolution Standardization. The original dimension of the images was 800x600 pixels, but both images were resized to 128x128. This dimension matches the input requirements of the neural network architecture.

Mask Processing. Masks represent the ground truth for the segmentation tasks. Mask processing requires extracting the relevant features, which involves reducing the color channels to a single channel that captures the essential segmentation information. After this, each RGB image is paired with its correct mask.

Optimization. The dataset undergoes a randomization process to prevent the model from memorizing the sequence of the data, which is crucial for avoiding potential biases. This step ensures that each training iteration presents the model with a diverse and representative sample of the data. Furthermore, batch processing contains a specified number of image-mask pairs that the model processes together during a single iteration.

Model Selection

This section gives a detailed description of the semantic segmentation models selected for this study. The main model is U-Net, but a comprehensive description of other architectures is provided.

Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are feed-forward neural networks composed of several feature maps extracted from an input. CNNs are suitable for grid data applications where fully connected neural networks may not be efficient because of computational power (speed time and memory) and spatial adjacency. These applications include image and video recognition, image classification, image segmentation, and natural language processing.

CNN comprises different layers: convolutions, pooling, and fully connected. The convolutional layer works by placing a filter or kernel (small matrix of numbers) over an array of image pixels, which creates a convolved feature map. This map is created by using convolutional multiplication which is element-wise with Equation (1).

$$x^{l-1} * w^{l} \tag{1}$$

Where x^{l-1} represents the input feature layer, and w^l is the kernel size. The summation of all layers plus a bias term, b_j^l, allows models to fit the data, and a non-linear operation shows the following equation for the 2D kernel with Equation (2).

$$x_j^l = \sigma\left(\sum_{i=l}^{N^{l-1}} x_i^{l-1} * w_{i,j}^l + b_j^l\right) \quad (2)$$

After a convolution operation, an activation function is applied elementwise (σ), illustrated in Equation (3).

$$\sigma\,(x) \;=\; max\{x, 0\} \quad (3)$$

By doing this, the system becomes non-linear, which enables the network to recognize intricate patterns. Pooling reduces the feature map's sample size, reducing the number of parameters. There are two main approaches for pooling: max pooling and average pooling. Max pooling is the most common method of pooling, which takes the maximum element of a feature map. On the other hand, average pooling calculates the average of the elements of the feature map. Finally, a fully connected layer is connected to all the activations in the previous layer so they can act as classifiers because they have enough information.

Fully Convolutional Networks

Fully convolutional networks (FCNs) are a class of CNNs designed explicitly for semantic segmentation. The main difference between these two networks is that FCNs replace the fully connected layers with convolutional layers. This modification allows them to process input images of any size and produce segmentation maps corresponding to the input image dimensions.

The architecture of FCNs consists of multiple layers designed for the segmentation process. First, FCNs use the same approach as an CNN to extract the image features. This approach starts with an input image, which can be any size. After this, the network uses multiple layers of filters (or kernels) to scan and extract the features of the image. As the filters move over the image, they extract important features like edges, textures, or specific shapes. Each layer captures fundamental features of the image little by little until deeper layers can identify more complex patterns. The creation of a feature map is the result of this feature extraction. A traditional CNN would use a fully connected layer to classify the image categories. However, an FCN replaces this with a 1x1 convolutional layer to convert the number of channels into classes. At this point, the feature map is smaller than the original image due to the convolution process. To return to the original size, FCN employs an upsampling layer, enlarging the feature map to the original size. During this process, labels are assigned to every pixel in the original image. The final output is a segmentation map matching the input image's original size. This segmentation map contains labeled features with a representative color. For instance, in an image of a street scene, pixels that are part of a building might be colored green. This detailed segmentation map divides the image into defined segments, each representing a different feature.

U-Net Architecture

U-Net is an extended FCN version initially designed for biomedical image segmentation. This network has been chosen as the primary resource for model training due to its segmentation accuracy and effectiveness in working with fewer images. The name of this network came from its particular 'U' shape. The architecture is formed by two main parts: the contraction path (also called downsampling) and the expansion path (also called upsampling). The left side of the architecture is the contraction path. This process is very similar to a typical CNN. It involves a

series of convolutional and pooling layers. Further details on each stage of U-Net architecture are provided in the following parts.

Encoding Blocks. Each encoding block in the U-Net architecture follows a sequence of two convolutional layers, each followed by batch normalization and a rectified linear unit (ReLU) activation. The convolutional layers use a 3x3 kernel with the same padding to preserve spatial dimensions and the He Normal kernel initializers for robust weight initialization. Batch normalization is employed after each convolution to stabilize learning and normalize the inputs to each activation layer. The ReLU activation function introduces non-linearity, enabling the network to learn complex patterns in the data. After each sequence of convolutions, a skip connection is created by setting aside the output of the second ReLU activation. This output is also potentially downsampled using a 2x2 max pooling operation, reducing the spatial dimensions and increasing the receptive field of the convolutional layers. The downsampled output is then passed to the following encoding block that serves as the input to the corresponding decoding block.

Decoding Block. This part merges the skip-connection input with the previous layer, processes it, and then returns an output. The decoding block begins with a transposed convolution (Conv2DTranspose) that upsamples the feature map and halves the number of filters, followed by a concatenation with the corresponding skip connection from the encoding path. This concatenation ensures that the high-resolution features from the encoding path are combined with the upsampled features to enable precise localization. After the merge operation, the combined feature map undergoes two more convolutional operations, each followed by batch normalization and ReLU activation, similar to the encoding block.

Final Assembly. The model follows a contracting path to capture context and an expanding path that enables precise localization, forming the 'U' shape of U-Net. After the final decoding block, a 3x3 Conv2D layer with ReLU activation is applied, followed by a 1x1 Conv2D layer with a sigmoid activation to map the output to the desired number of classes for pixel-wise classification. In the final assembly of the U-Net model, the input layer takes the shape of the input data, followed by successive encoding blocks that reduce the dimensionality while increasing the depth of the feature maps. The lowest level of the U-shape is the bridge between the encoding and decoding paths, where the feature map is at its most abstract representation. The decoding path then progressively recovers spatial resolution, combining the abstracted features with the detailed spatial information from the skip connections. The final layer of the network uses a sigmoid activation function to produce the final segmentation map, indicating the class probabilities for each pixel.

The U-Net model is defined by a function that takes the input size of the original image, number of filters, and number of classes as parameters, allowing for flexibility and adaptability of the network architecture to various input sizes and types of segmentation tasks. The detailed steps of the U-Net model's implementation process have been abstracted into a high-level pseudocode, which can be found in Appendix A. This pseudocode provides a clear and concise representation of the sequence of operations within the U-Net architecture, further illustrating the processes described in this section.

Model Training

This section describes the training procedure undertaken for the semantic segmentation model. The model training process starts with finding hyperparameter optimization for the model and culminates in the training of the final model with the identified best parameters.

Hyperparameter Optimization

The initial training phase involves an extensive review of what hyperparameters authors have used in past studies. This study focuses on tuning three hyperparameters: learning rate, epochs, and batch size. Table 3 shows the literature review in past studies for the selected hyperparameters of this study.

Informed by the literature, a preliminary set of hyperparameters is chosen as a baseline. Because of this, the initial hyperparameter space for the random search method is set to a learning rate in the range of -5 to -1 with a logarithmic base, epochs space is set from 10 to 101, and batch size space is set from 16 to 256. The random search is executed over a defined number of iterations (five). The best-performing set from this phase, which is determined by the highest validation accuracy, provides a starting point for the grid search. The best hyperparameters are learning rate equal to 0.00233, 98 epochs, and a batch size of 32. These numbers define the grid set. This method involves a more targeted range of values, allowing a precise calibration of hyperparameters. The hyperparameter optimization concludes with the best hyperparameters (learning rate equals 0.0025, 32 epochs, and a batch size of 98), which are the input of the final training, and it is described in the following section.

Table 3: Hyperparameters for segmentation models obtained from literature review

		Hyperparameters		
ID #	Method	Learning Rate	Batch Size	Epochs
RP1	SSD	0.004	10	4000
RP2	SSD	0.004	10	4000
	Faster RCNN	0.0002	1	7000
RP3	YOLOv4	0.001	8	50
		1e-4	4	500
	CenterNet	1.25e-4	16	600
	Faster RCNN	0.01	1	400
RP4	Stereo RCNN	0.001, reduced to 0.1	512	20
RP5	YOLOv4	0.0013	4000	2000
RP6	UNet	5e-4	10	50
RP7	DeepLabv3+	5e-3	2	50
RP8	---	NA	NA	NA
RP9	SPNet; FCN; SegNet	NA	NA	100
Random Search	**U-Net**	**0.00233**	**32**	**98**
Grid Search	**U-Net**	**0.0025**	**32**	**95**
Final	**U-Net**	**0.0025**	**32**	**95**

Final Model Training

Following the rigorous hyperparameter optimization, the optimal hyperparameters are employed in the final training stage. The optimizer chosen is Adam, a popular choice for deep learning tasks due to its adaptive learning rate capabilities. The loss function of sparse categorical cross-entropy is chosen for its suitability in multi-class classification problems. The optimal hyperparameters, loss function, and optimizer go into a compilation step. This step is a critical phase in preparing the neural network for training.

After this, the model is fit to the data, with the training and validation datasets appropriately assigned. The training is executed over the optimal number of epochs, with a batch size determined to be the most effective from the hyperparameter tuning stage. A callback setup

is employed to ensure effective learning and to avoid overfitting. This setup consists of an early stopping and a reduced learning rate, and they are passed to the function. The data is shuffled during training, which is a good practice to prevent the model from learning any accidental patterns from the order of the data.

In conclusion, the model combines strategies for preventing overfitting, optimizing learning, and monitoring performance with the validation accuracy metric. The model training process was not just about fitting the model to the data but ensuring that it could generalize well to new, unseen data.

Evaluation Metrics

The evaluation of this model is judged by the evaluation metrics that reflect the model's relevance in autonomous driving applications. The primary evaluation metric is accuracy (Equation (4)), which reflects the overall proportion of correctly classified pixels in the segmentation maps compared to the ground truth.

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Neg}}{\text{True Positive} + \text{False Positive} + \text{True Positive} + \text{False Neg}} \quad (4)$$

Accuracy alone might not be a complete measure because of the possible class imbalance in the road scenes. Therefore, the model is evaluated by its loss function. The loss function quantifies the model's error in predictions. This study utilizes sparse categorical cross-entropy, which computes the loss between the labels and predictions. It is a type of probabilistic loss, and it is chosen for its efficiency in handling classification problems with multiple classes that are mutually exclusive. A lower value of loss function indicates good performance. This metric is vital for safety-critical applications since it shows the difference between the predicted probabilities and the actual distribution of classes.

The performance of a semantic segmentation model cannot solely rely on accuracy and loss function. A model that excels in maximizing its accuracy and minimizing its loss function could fail in real-world situations if it does not balance precision and recall. Recall, also called sensitivity or true positive rate (TDR), identifies true positives among all actual positives. In other words, recall measures the model's ability to identify all actual instances of an object. On the other hand, precision identifies true positives among all predicted positives. It ensures that when a model predicts an object's presence, that prediction is accurate. A balance between these two metrics is essential, especially for autonomous driving applications, which are highly safety-critical. For example, consider the crucial role of an autonomous vehicle's system in distinguishing road signs from other roadside objects. The system must identify every road sign correctly (achieving high recall) to comply with traffic rules and ensure passenger safety. At the same time, it is equally important to avoid misclassifying other objects as road signs (maintaining high precision) to prevent confusion and erratic driving behavior. If the system fails to maintain this delicate balance, the consequences could range from traffic violations to engaging in unnecessary and potentially hazardous maneuvers. For autonomous driving applications, it is expected that precision and recall values are equal to or greater than 90 %. Equations (5) and (6) illustrate recall and precision, respectively.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{Fase Positive}} \tag{5}$$

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{6}$$

Another key metric in this study is the IoU, which is a crucial measure in image segmentation tasks. This metric is illustrated in Equation (7). It quantifies the extent of overlap

between the predicted segmentation and the ground truth segments. On other words, this metric measures how much the model's predicted boundaries for objects in an image coincide with the actual. For autonomous driving, IoU values are expected to be 0.70 or higher. This level of precision is vital for autonomous vehicles that rely on exact and prompt interpretations of their environment to navigate complex and dynamic driving conditions safely. Complementing all these values is the F1-score, which combines precision and recall into a single metric, illustrated in Equation (8). A good trade-off between these two metrics is important for autonomous driving applications. Generally, an F1 score of 0.7 or higher is often considered good. The last metric used in this study is specificity. It indicates the ratio of true negatives among all actual negatives. In the context of autonomous driving, specificity concerns accurately identifying objects that are neither hazardous nor necessary when driving. A high-specificity model helps prevent unwarranted evasive maneuvers that could cause confusion or accidents on the road by ensuring that an autonomous vehicle will not mistakenly interpret benign scenarios as threats. A value close to 100 % is crucial for autonomous driving applications.

$$\mathrm{IoU} = \frac{\mathrm{Target} \cap \mathrm{Predicted}}{\mathrm{Target} \cup \mathrm{Predicted}} \tag{7}$$

$$\mathrm{F1-score} = 2 * \frac{(\mathrm{Precision} * \mathrm{Recall})}{(\mathrm{Precision} + \mathrm{Recall})} \tag{8}$$

Experimental Setup

This section clarifies the experimental framework (hardware and software specifications) used in developing and evaluating the semantic segmentation model.

Hardware Configuration

The training and testing of the model are conducted on equipment with a 12th Gen Intel i7 processor, 16 GB of RAM, and NVIDIA GeForce RTX 3060 GPU with 12 GB. The choice of hardware is driven by the need for high computational power and memory bandwidth to handle the large datasets typical of autonomous driving scenarios and to expedite the training process with GPU acceleration. This GPU was selected for its ability to perform parallel processing, a critical feature for deep learning algorithms. The VRAM allows for handling large neural networks and datasets, a common characteristic in autonomous driving scenarios, and the advanced GPU architecture significantly accelerates the training and inference processes.

Software Environment

The experiment is run within a Windows Operating System environment. The specific library used for this model is TensorFlow, chosen for its robustness and extensive library support. The programming is primarily conducted in Jupyter Notebook with Python 3.9. The additional libraries utilized in this study are Numpy for numerical processing, Matplotlib for visualization, Pandas for data manipulation, ImageIO for image reading, Scikit-Learn for machine learning classification and preprocessing data, and Keras for training deep learning models.

In the setup of TensorFlow, the CUDA toolkit version 11.2.2 is installed along with cuDNN library version 8.10.77. These versions are compatible with the Windows Operating System and the installed CUDA version of the machine. CUDA serves as the underlying layer that allows direct access to the GPU's virtual instruction set and parallel computational elements, essential for efficiently running complex deep learning algorithms. The required TensorFlow

version is 2.9, compatible with the selected CUDA version. All these libraries are installed in an Anaconda virtual environment to avoid damage to the central system.

In conclusion, the configuration of hardware and software environments plays a crucial role in the development of the semantic segmentation model. The chosen setup reflects a thoughtful balance between computational capability and efficiency, ensuring that the model is trained within a robust framework and subjected to rigorous testing that mimics real-world conditions. Full details are provided to facilitate replication of the study, underscoring the commitment to transparency and rigor in this research.

CHAPTER IV

RESULTS

This chapter introduces the results obtained from the implementation and evaluation of the semantic segmentation model designed for autonomous driving perception. This section of the thesis analyzes the model's performance and highlights its effectiveness and reliability, building on the evaluation metrics presented in Chapter 3. The findings provide insight into the algorithm's potential to improve AV perception systems, decision-making processes, and safety on the road.

Model Performance

Accuracy and Loss Function

The main metrics of evaluation are accuracy and loss function. Figure 2 shows the model's accuracy. The blue line represents the training accuracy, and the orange line represents the validation accuracy. Both accuracy measures show a steep increase within the initial epochs, rapidly approaching a plateau near 100%. This rapid convergence suggests the model's ability to learn from the training data quickly. The close alignment between training and validation accuracy indicates that the model generalizes well to unseen data, a crucial factor for real-world deployment in autonomous vehicles. The graph is configured to display an axis limited to the

first 30 epochs of training. While the original configuration extended up to 95 epochs, the model achieves peak performance quickly, suggesting limited benefits from additional training epochs. This observation implies that the model has sufficient robustness, reducing the need for prolonged training periods.

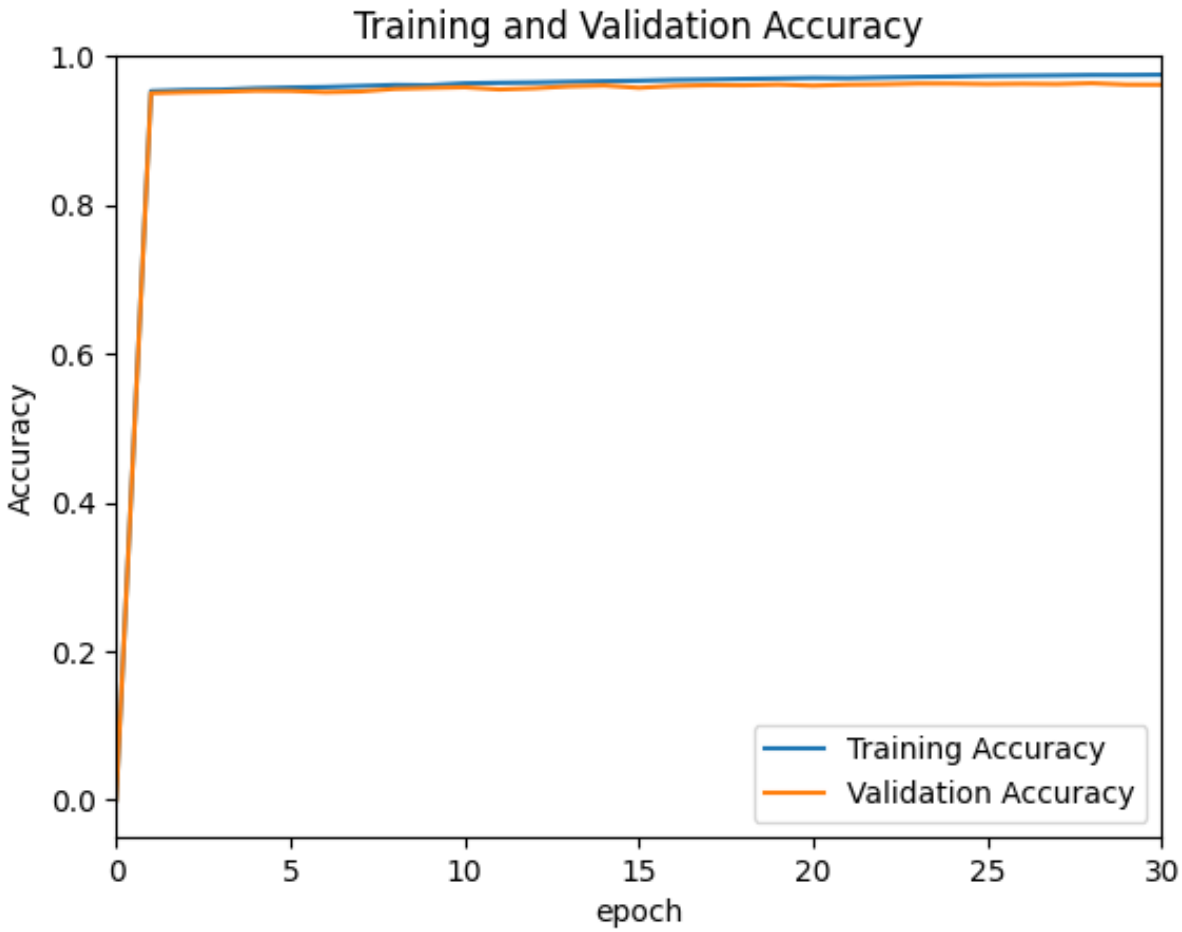

Figure 2: Accuracy Plot for training and validation datasets

The loss function of the model is evaluated in Figure 3. This figure illustrates a decline in training loss during the early stage, suggesting that the model learns quickly and effectively from the training data. The small gap between the training and validation loss strongly indicates the model's ability to generalize. An early stopping stops this gap. Suppose the validation dataset

starts to diverge from the training set, the early stopping actives, and stop the training. The minor fluctuations observed in the validation loss are typical and can be attributed to the variability inherent in the validation dataset. These fluctuations do not indicate model instability, as they do not demonstrate a consistent upward trend or substantial spikes.

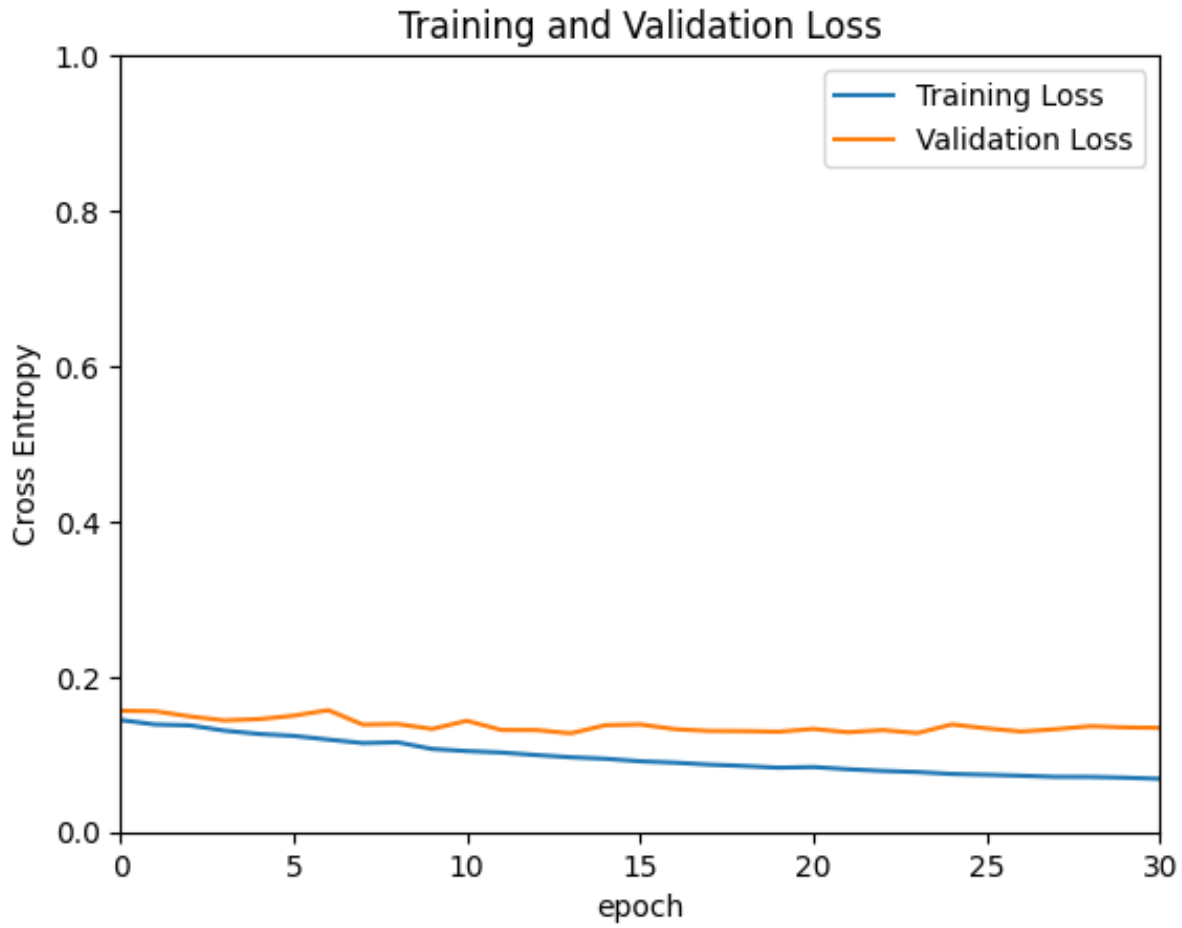

Figure 3: Loss Function plot for training and validation datasets

Supplementary Evaluation Metrics

Since model accuracy alone may not always be enough to determine whether a model is optimal, precision and recall are the commonly chosen metrics used in addition to accuracy to judge model performance. In addition, this model is rated specific, IoU, and F1-score as

supplementary metrics for model performance in all datasets (training, validation, and test datasets).

Training Data Performance. Table 4 outlines the model's performance on the training data. The model exhibits strong recall and precision across most classes, with high road, sidewalk, car, and building scores. The unlabeled class shows excellent results, representing areas not covered by other classes. The IoU and F1 scores are notably high for these classes, demonstrating the model's precise segmentation capabilities. The classes with the lowest scores are traffic signs, poles, and fences. This has to deal with the limited training data containing these features. Overall, the training dataset shows excellent performance, with numbers that are higher than or equal to the metrics threshold.

Table 4: Evaluation metrics of the training images

Class	Recall	Precision	Specificity	IoU	F1-Score
All Classes	0.91	0.94	1.0	0.87	0.92
Unlabeled	0.99	0.99	0.99	0.97	0.99
Building	0.99	0.98	1.0	0.97	0.98
Fence	0.73	0.85	1.0	0.65	0.79
Other	0.83	0.9	1.0	0.76	0.86
Pedestrian	0.86	0.93	1.0	0.81	0.89
Pole	0.86	0.93	1.0	0.81	0.89
Road Line	0.93	0.92	1.0	0.86	0.92
Road	1.0	1.0	1.0	0.99	1.0
Sidewalk	0.99	0.98	1.0	0.97	0.98
Vegetation	0.95	0.94	0.99	0.9	0.94
Car	0.99	0.98	1.0	0.97	0.98
Wall	0.94	0.93	1.0	0.88	0.93
Traffic Sign	0.8	0.89	1.0	0.73	0.84

Validation Data Performance. Table 5 shows the performance of the validation data. Performance metrics are anticipated to decline as the model is evaluated on unseen. However, the model maintains high precision and specificity, indicating a solid predictive performance

with low false-positive rates. The car and sidewalk classes continue to show high accuracy, while classes like pedestrian, pole and traffic sign present opportunities for improvement, as indicated by their lower recall and IoU scores.

Table 5: Evaluation metrics of the validation images

Class	Recall	Precision	Specificity	IoU	F1-Score
All Classes	0.82	0.87	1.0	0.75	0.84
Unlabeled	0.98	0.98	0.99	0.96	0.98
Building	0.98	0.97	1.0	0.94	0.97
Fence	0.68	0.79	1.0	0.58	0.73
Other	0.72	0.83	1.0	0.63	0.77
Pedestrian	0.62	0.83	1.0	0.55	0.71
Pole	0.62	0.75	1.0	0.51	0.68
Road Line	0.7	0.71	1.0	0.54	0.7
Road	0.99	0.99	0.99	0.97	0.99
Sidewalk	0.96	0.95	1.0	0.92	0.95
Vegetation	0.94	0.92	0.99	0.86	0.93
Car	0.98	0.98	1.0	0.96	0.98
Wall	0.9	0.89	1.0	0.81	0.89
Traffic Sign	0.61	0.76	1.0	0.51	0.68

Test Data Performance. The testing data results are presented in Table 6. The model shows consistency in the road, sidewalk, and car classes. These classes show high precision and IoU scores similar to training and validation scores. This consistency is critical, reflecting the model's ability to maintain high segmentation standards even on entirely new data. Lower scores in classes such as pedestrian and traffic signal suggest areas where the model might benefit from additional training data or algorithmic refinement.

Table 6: Evaluation metrics of the test images

Class	Recall	Precision	Specificity	IoU	F1-Score
All Classes	0.82	0.87	1.0	0.75	0.84
Unlabeled	0.98	0.98	0.99	0.97	0.98
Building	0.98	0.96	1.0	0.94	0.97
Fence	0.68	0.8	1.0	0.58	0.74
Other	0.69	0.81	1.0	0.6	0.75
Pedestrian	0.67	0.83	1.0	0.59	0.74
Pole	0.61	0.74	1.0	0.5	0.67
Road Line	0.69	0.7	1.0	0.53	0.69
Road	0.99	0.98	0.99	0.97	0.98
Sidewalk	0.96	0.95	1.0	0.92	0.95
Vegetation	0.94	0.92	0.99	0.87	0.93
Car	0.98	0.98	1.0	0.96	0.98
Wall	0.9	0.89	1.0	0.81	0.89
Traffic Sign	0.62	0.76	1.0	0.52	0.68

The average results are shown in Table 7. The results indicate that the model performs consistently across the three datasets with slightly better performance on the training set than the validation and test sets. The model demonstrates high accuracy (97%-98%) across all datasets, indicating that most predictions are correct. However, there is a slight drop in recall, precision, IoU, and F1-Score when moving from training to validation and test datasets, which could suggest the model is slightly overfitting to the training data. Still, the differences need to be more significant to indicate insufficient generalization capabilities. The specificity is perfect at 100%, indicating that the model is excellent at identifying negative cases. Overall, the results are promising, but there may be room for improvement, particularly in enhancing the model's recall and precision on the validation and test datasets.

Table 7: Average results for each metric in distinct datasets

Dataset	Model Accuracy	Mean Recall	Mean Precision	Mean Specificity	Mean IoU	Mean F1-Score
Training	98.24%	91%	94%	100%	87%	92%
Validation	96.64%	82%	87%	100%	75%	84%
Test	96.6%	82%	87%	100%	75%	84%

Image Segmentation Evaluations

Image segmentation evaluations are crucial in validating the performance of a segmentation model. These evaluations compare the predicted segmentation (predicted mask) with the ground truth data (true mask) across various datasets. The input image is the original, unaltered image. This image is what the model will attempt to segment. The true mask is the pre-annotated ground truth segmentation. This mask delineates the features in the image. Finally, the predicted mask is the output from the segmentation model. It represents the model's attempt to replicate the true mask based on what it has learned during training.

Training Data Performance

Figure 4 shows the performance of the model on the training data. The visual inspection of the segmentation output reveals that the model has achieved a high level of proficiency in recognizing and segmenting larger distinct objects within the urban landscape. The color consistency between the true and predicted masks indicates the model's accurate classification capabilities, which form the backbone of reliable segmentation. The analysis also uncovers some challenges faced by the model. The boundary delineation around smaller or more complex objects is less clear than the true mask. This limitation suggests a difficulty in the model's edge detection capabilities, which is crucial for high-fidelity segmentation tasks. To address this challenge, the model may benefit from increasing the resolution of the input images during training. Unfortunately, the hardware could not handle higher resolution due to a memory issue with the GPU. This challenge can be fixed by upgrading the hardware, and it does not affect the main goal of this study.

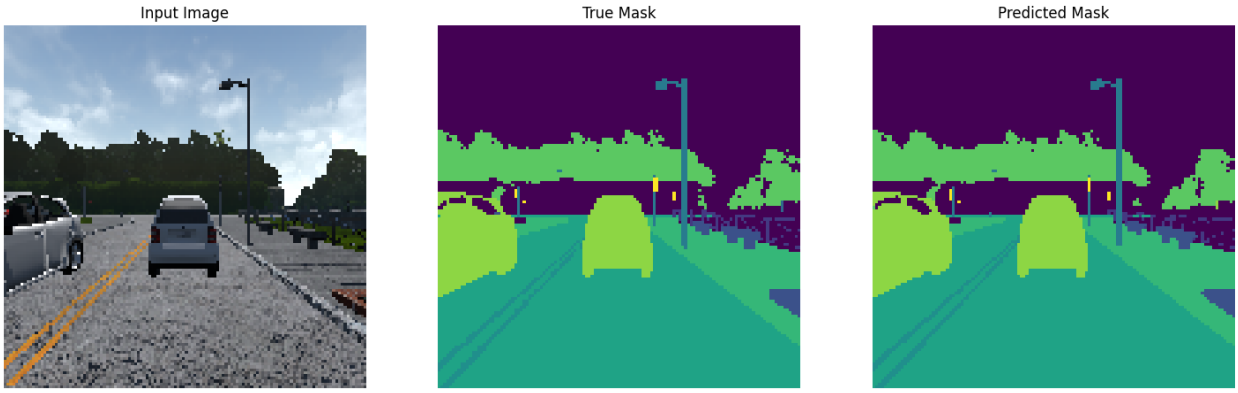

Figure 4: Predict and compare masks of images in the training set

Validation Data Performance

Figure 5 shows the performance of the validation data. The predicted mask mirrors the true mask, indicating that the model has generalized well from the training data to the validation data. This suggests that the features learned during training are robust and transferable to new, unseen images. The predicted mask output demonstrates the model's capacity to identify and segment various features within an urban scene. For instance, the car is identifiable with its respective colored label. However, there are observable discrepancies at the edges, and the model has not perfectly captured the contour of the car. This flaw can be easily fixed by increasing the image resolution, like the situation from the training data. In the predicted mask, it can be observed that pedestrians are segmented with a different color in the true mask, emphasizing their importance as dynamic and critical objects in urban settings.

In summary, the model demonstrates a competent level of segmentation for large and distinct objects. It requires a small refinement in handling smaller objects, such as boundary delineation. However, similar to the training data, this situation can be easily fixed by increasing the dimension of the input image.

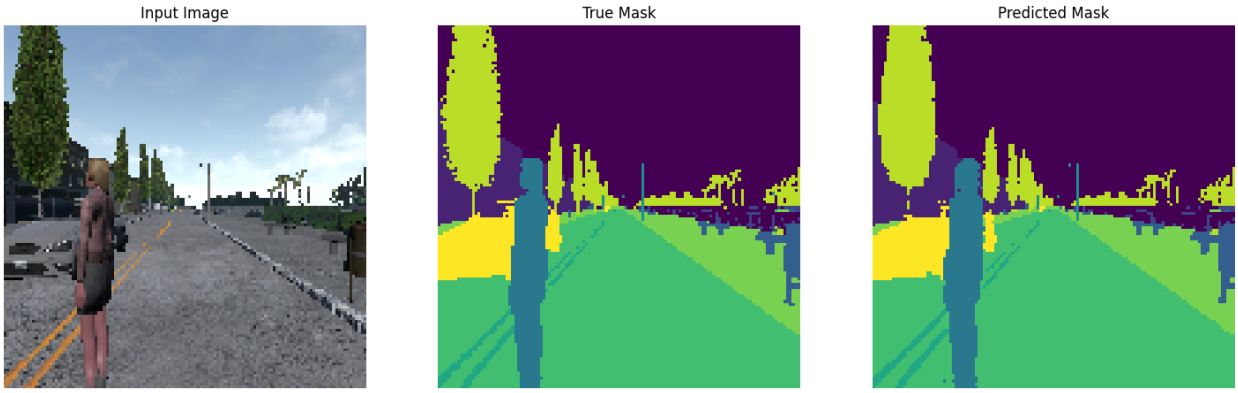

Figure 5: Predict and compare masks of images in the validation set

Test Data Performance

Figure 6 provides evaluation results for the test data. The predicted mask presents the model's ability to generalize and apply its learned segmentation to new, unseen data, which is essential for determining its real-world applicability. In the output image, the pedestrian is recognized by its label color. There is room for improvement because the predicted mask exhibits minimal blending at the boundaries of the background. Beyond the pedestrian segmentation, the performance of other urban landscape elements within the predicted mask indicates a solid understanding of the larger, more uniform segments, such as the road surface and building facades.

Figure 6: Predict and compare masks of images in the test set

The semantic segmentation model discussed in this chapter shows impressive capabilities in autonomous driving perception. Its rapid learning curve is evident from the high accuracy and low loss demonstrated in early training epochs, with near-perfect performance in classifying roads, sidewalks, and vehicles. Despite slight declines in performance on validation and test datasets, the model maintains strong generalization, indicating its potential for real-world application. Visual evaluations further confirm its proficiency in segmenting larger urban features. However, it faces challenges with smaller or complex objects, a limitation that could be mitigated with improved image resolution or hardware upgrades. Overall, this model stands out for its robustness and accuracy in autonomous vehicle perception tasks.

CHAPTER V

CONCLUSION

The research presented in this thesis offers a comprehensive analysis and implementation of a U-Net-based semantic segmentation model for enhancing autonomous vehicle perception systems. The main objective of this thesis is to demonstrate the feasibility and effectiveness of using a deep learning architecture to accurately identify and classify various elements within urban driving environments, which is critical for the development of safe and reliable autonomous vehicles.

This chapter provides a summary of the model, a review of the study design, an examination of limitations, and a summary of major findings, along with conclusions and recommendations for further studies.

Summary of Findings

This study aims to improve autonomous car perception systems by applying a semantic segmentation model based on U-Net. The study explores the model's training, optimization, and performance across various urban scene elements, revealing an exemplary level of accuracy in segmenting roads, sidewalks, vehicles, and pedestrians. The consistent performance across training, validation, and test datasets underscores the model's robustness and generalizability. The results demonstrate that the model achieves impressive accuracy, with the ability to learn

and generalize well to unseen data. The training dataset exhibits superior performance, with the validation and test datasets following closely, indicating the model's potential for real-world application. However, certain classes, such as traffic signs and poles, presented challenges, suggesting a need for further model refinement. The results highlight the model's remarkable accuracy, reaching up to 97% on the training set and consistently maintaining high performance across validation and test datasets. The model's precision and specificity, essential for the safe navigation of autonomous vehicles in dynamic urban environments, are particularly impressive.

Conclusion and Implications

The study concludes that deep learning methods are viable and effective approaches for semantic segmentation in autonomous driving applications. The U-Net model showcases impressive segmentation accuracy, with IoU scores exceeding 0.95 for several classes, significantly higher than some existing models like ENet, SegNet, and other versions of FCNs tested on similar datasets.

The model demonstrates remarkable precision and specificity, which are crucial for the accurate navigation of autonomous vehicles. The ability to distinguish between different elements of urban landscapes, such as roads, sidewalks, and cars, with high precision and recall rates underscores the model’s potential in real-world scenarios. However, it was observed that the segmentation of smaller objects, like traffic signs and poles, needed to be more accurate, possibly due to the limited resolution of the input images.

Despite these limitations, the consistency in performance across training, validation, and test datasets speaks to the model's generalizability. This consistency is a key advantage, as models that perform well only on training data but fail to generalize are of limited practical use.

Moreover, the high specificity scores across all datasets indicate the model's proficiency in correctly identifying negative cases, which is vital for the safety of autonomous vehicles to avoid unnecessary or hazardous maneuvers.

In conclusion, the U-Net-based semantic segmentation model developed and analyzed in this thesis represents a significant step forward in autonomous vehicle perception. Its ability to accurately interpret complex urban scenes can reduce the likelihood of accidents and improve the overall safety of autonomous navigation. However, the challenges identified in segmenting smaller objects and needing higher-resolution images point to areas where further improvements are necessary. Addressing these limitations will be important in advancing the field and ensuring that autonomous vehicles can operate reliably in all driving conditions.

Limitations

Despite the high accuracy, certain challenges were identified, such as the model's edge detection capabilities and handling smaller or more complex objects. Due to hardware constraints, these issues are attributed to the limited resolution of input images. However, the model demonstrates robust generalization abilities, with minimal performance drops when transitioning from training to validation and test datasets, which indicates its real-world applicability.

Recommendations

The research recommends an increased resolution of input images, where hardware capabilities allow for improved model accuracy, especially in edge detection and segmentation of smaller objects. Further, it suggests incorporating multimodal sensor data to enhance perception accuracy.

Recommendations for Future Work

Image Cleaning for Overfitting Prevention

Future work should focus on developing algorithms that can evaluate the similarity between images within the dataset, thereby identifying and eliminating redundant or highly similar instances that may contribute to overfitting. Ensuring a diverse and representative training dataset could significantly improve the model's ability to generalize to new, unseen data.

Multimodal Sensor Fusion

Future models should incorporate a multimodal sensor fusion framework. This approach would combine data from various sensor types, such as LiDAR, radar, and cameras, to comprehensively understand the vehicle's surroundings. Training the semantic segmentation model to utilize this fused data effectively could enhance the perception system's accuracy, especially in challenging visibility conditions.

Semantic Instance Segmentation

Beyond semantic segmentation, future work could also look into instance segmentation, which involves identifying each instance of a particular object class separately. Instance segmentation would enable the perception system to count and track multiple objects of the same class individually, such as distinguishing between two cars of the same make and model.

The recommendations provide a roadmap for advancing autonomous vehicle technology, emphasizing the importance of machine learning in improving safety and efficiency. This thesis makes a significant contribution to this dynamic field, underlining the necessity of continuous innovation and research.

REFERENCES

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(12), 2481–2495. https://doi.org/10.1109/tpami.2016.2644615

Berlincioni, L., Becattini, F., Galteri, L., Seidenari, L., & Bimbo, A. D. (2019). Road layout understanding by generative adversarial inpainting. *Inpainting and Denoising Challenges*, 111–128. https://doi.org/10.1007/978-3-030-25614-2_10

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2014). DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40*(4), 834–848. https://doi.org/10.1109/tpami.2017.2699184

Chen, L.-C., Papandreou, G., Schroff, F., & Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation.

Dosovitskiy, A., Ros G., Codevilla F., Lopez A., & Koltun V. (2017). CARLA: An open urban driving simulator, *Proc. Conf. Robot Learn*, 1-16.

Gao, W., Tang, J., & Wang, T. (2021). An object detection research method based on Carla Simulation. *Journal of Physics: Conference Series, 1948*(1), 012163. https://doi.org/10.1088/1742-6596/1948/1/012163

Girshick, R. (2015). Fast R-CNN. *2015 IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/iccv.2015.169

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/cvpr.2014.81

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/iccv.2017.322

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional Neural Networks. *Communications of the ACM, 60*(6), 84–90. https://doi.org/10.1145/3065386

Kuutti, S., Bowden, R., Jin, Y., Barber, P., & Fallah, S. (2021). A survey of deep learning applications to Autonomous Vehicle Control. *IEEE Transactions on Intelligent Transportation Systems*, *22*(2), 712–733. https://doi.org/10.1109/tits.2019.2962338

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single-shot MultiBox detector. *Computer Vision – ECCV 2016*, 21–37. https://doi.org/10.1007/978-3-319-46448-0_2

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2015.7298965

Martínez-Díaz, M., & Soriguera, F. (2018). Autonomous vehicles: Theoretical and practical challenges. *Transportation Research Procedia*, *33*, 275–282. https://doi.org/10.1016/j.trpro.2018.10.103

Niranjan, D. R., VinayKarthik, B. C., & Mohana. (2021). Deep learning-based object detection model for autonomous driving research using Carla Simulator. *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*. https://doi.org/10.1109/icosec51865.2021.9591747

Niranjan, D., VinayKarthik, B., & Mohana. (2021). Performance analysis of SSD and faster RCNN multi-class object detection model for autonomous driving vehicle research using Carla Simulator. *2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. https://doi.org/10.1109/icecct52121.2021.9616712

Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. *2015 IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/10.1109/iccv.2015.178

Paszke A., Chaurasia A., Kim S. & Culurciello E. (2016). ENet: A deep neural network architecture for real-time semantic segmentation.

Redmon, J., & Farhadi, A. (2017). Yolo9000: Better, faster, stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2016.91

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement.

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/tpami.2016.2577031

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Lecture Notes in Computer Science*, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

Singh, S. (2015). Critical reasons for crashes investigated in the National Motor Vehicle Crash Causation Survey. *Nature Highway Traffic Safety Admin., U.S. Nat. Center Statist. Anal., Tech. Rep. DOT HS 812,* 115, 1–2

Simonyan, K. & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *International Conference of Learning Representations (ICLR).*

Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2015.7298594

Talebpour, A., & Mahmassani, H. S. (2016). Influence of connected and autonomous vehicles on traffic flow stability and throughput. *Transportation Research Part C: Emerging Technologies*, *71*, 143–163. https://doi.org/10.1016/j.trc.2016.07.007

Treml, M., Arjona-Medina, J. A., Unterthiner, T., Durgesh, R., Friedmann, F., Schuberth, P., Hochreiter, S., et al. (2016). Speeding up semantic segmentation for autonomous driving.

Yaqoob, I., Khan, L. U., Kazmi, S. M., Imran, M., Guizani, N., & Hong, C. S. (2020). Autonomous driving cars in smart cities: Recent advances, requirements, and challenges. *IEEE Network*, *34*(1), 174–181. https://doi.org/10.1109/mnet.2019.1900120

APPENDIX A

APPENDIX A

PSEUDOCODE OF U-NET SEMANTIC SEGMENTATION

The following pseudocode, illustrated in Figure 7, is a structured representation of the U-Net architecture for image segmentation. It abstracts the Python code into a conceptual algorithm, encapsulating the U-Net model's essential components and steps. This pseudocode includes setting up the environment, preparing the data, defining the model structure, and executing the training process.

```
Algorithm U-Net Image Segmentation
begin procedure
    Set TensorFlow log level to '3'

    Define paths for image and mask directories (string)

    Define function list_image_paths(directory: string) -> list of file paths (string)

    Get image_paths and mask_paths (lists of strings)
    Print count of images and masks (integer)

    For 3 random samples (index: integer)
        Read image (size: [height, width, 3]) and mask (size: [height, width, 1 or 3])
        If mask has 3 channels, convert to 1 channel
        Display image and mask

    Split paths into train, validation, test sets (70%, 15%, 15%)
    Print set sizes (integer)

    Define function read_image (returns tensors of size [128, 128, channels])
    Define function data_generator (returns TensorFlow dataset)

    Create datasets for training, validation, test (batch_size: integer)

    Define encoding_block and decoding_block functions for U-Net (return tensors)
    Define unet_model function (returns compiled TensorFlow model)

    Set model training callbacks for early stopping and learning rate reduction
    Train model (learning_rate: integer, epochs: integer, batch_size: integer)

end procedure
```

Figure 7. Algorithm U-Net Image Segmentation

BIOGRAPHICAL SKETCH

Oscar Gilberto De Leon Vazquez was born and raised in Mexico on November 1, 1997. He moved to the United States in October 2016. He started college at the University of Texas Rio Grande Valley in January 2018. During his undergraduate career, Oscar participated in the Society of Hispanic Professional Engineers (SHPE) as a senior advisor and in the American Society of Mechanical Engineers (ASME) as team co-lead for design competitions. Additionally, Oscar interned with Micron Technology in the Key Equipment Group (KEG) as a Test Solutions Engineer in Summer 2021. Oscar earned his bachelor's degree in mechanical engineering in December 2021. He began his graduate studies in January 2022 at the National Science Foundation CREST Center for Multidisciplinary Research Excellence in Cyber-physical Infrastructure Systems (MECIS), where the National Science Foundation funds his research. Oscar interned with Cummins in the Research and Development department during his graduate studies as a Model-Based Systems Engineer in Summer 2023. Oscar was recognized as a scholar in the Great Mind in STEM (GMiS) program in October 2022. He graduated in December 2023 from UTRGV with a Master's of Science in Mechanical Engineering and planned to take a role as a Manufacturing Engineer at the Boeing Company, with aspirations to complete a PhD in Mechanical Engineering. He can be contacted by email at oscar.deleon.engineer@gmail.com.