DEEP LEARNING-BASED SEMANTIC SEGMENTATION IN AUTONOMOUS DRIVING

by

Hrag – Harout Jebamikyous
Bachelor of Engineering Technology, Yorkville University, 2018

An MRP

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Engineering

in the program of Electrical and Computer Engineering

Toronto, Ontario, Canada, 2021

Author's Declaration for Electronic Submission of an MRP

I hereby declare that I am the sole author of this MRP. This is a true copy of the MRP, including any required final revisions.

I authorize Ryerson University to lend this MRP to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this MRP by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my MRP may be made electronically available to the public.

DEEP LEARNING-BASED SEMANTIC SEGMENTATION IN AUTONOMOUS DRIVING

Hrag – Harout Jebamikyous
Master of Engineering in Electrical and Computer Engineering
Ryerson University, 2021

ABSTRACT

Perception is a fundamental task of autonomous driving systems, which gathers all the necessary information about the surrounding environment of the moving vehicle. Then a decision-making system takes the perception data as input and provides the optimum decision given a scenario, which maximizes the safety of the passengers. In this project, we have developed variants of the U-Net model to perform semantic segmentation on urban scene images to understand the surroundings of an autonomous vehicle. The U-Net model and its variants are adopted for semantic segmentation in this project to account for the power of the UNet in handling large and small datasets. We have also compared the best-performing variant with other commonly used semantic segmentation models. The comparative analysis was performed using three well-known models, including FCN-16, FCN-8, and SegNet. After conducting sensitivity and comparative analysis, it is concluded that the U-Net variants performed the best in terms of the Intersection over Union (IoU) evaluation metric and other quality metrics.

Acknowledgements

Throughout the writing of this thesis, I have received a great deal of support and assistance.

I would first like to thank my supervisor, Professor Rasha Kashef, whose expertise was invaluable in formulating the thesis surveys and methodology. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.

I would particularly like to single out my undergraduate professor, Dr. Ahmad Ibrahim for his valuable guidance throughout my studies. He has provided me with the tools that I needed to choose the right direction and successfully complete my master's degree.

In addition, I would like to thank my parents and my sister for their wise counsel and sympathetic ear. You were always there for me. Finally, I could not have completed this degree without the support of my cousin Njteh Ordekian, who provided stimulating discussions as well as happy distractions to rest my mind outside of my study zone.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AVP – Autonomous Vehicle Perception

ADAS – Advanced Driver Assistant Systems

SSD – Single Shot Detector

CNN – Convolutional Neural Network

FCN – Fully Convolutional Network

ENET – Efficient Net

DNN – Deep Neural Network

MLP – Multilayer Perceptron

ROI – Region of Interest

ReLU – Rectified Linear Unit

Chapter 1: Introduction

As technology constantly evolves, autonomous vehicles are becoming more popular, accessible, and affordable for more people in different countries and from different economic classes. Increasing accessibility results in a safer transportation experience, fewer deaths, and minimal injuries due to human-made mistakes that cause catastrophic accidents. To ensure the safety of individuals, it is necessary to deploy highly efficient and accurate learning models trained on a broad range of driving scenarios to precisely detect the surrounding objects under different weather and lighting conditions. This learning procedure via training will adjust the vehicle's decision-making process and control mechanism to take the necessary actions.

Autonomous Vehicle Perception (AVP) in driving systems collects the necessary information about the surrounding environment of the moving vehicle. The perception data is then fed to a learning model to obtain an optimum decision. The two main methods used in the perception of autonomous vehicles: Semantic Segmentation and Object detection; both tasks work primarily with images. Semantic segmentation is the process of assigning each pixel in an image to a particular class. These class labels could be a person, bicycle, tree, etc. Semantic segmentation is considered an image classification task at a pixel level. Object detection is the task of identifying and locating an object of interest in an image and draw a bounding box around that object.

In this paper, we tackled the problem of semantic segmentation using a very well-known semantic segmentation model used for biomedical image segmentation tasks, called U-Net. The name of the model is inspired by the shape of the architecture, which looks like the letter U. The U-Net model is one of the few existing architectures which perform well on small datasets and was not previously tested in an autonomous driving scenario with a large number of classes, and a small number of training images. After training multiple U-Net models with different

activation functions, regularization techniques, and different depths, we proved that U-Net could have a promising future in the field of autonomous driving and scene understanding due to its ability to answer the “What” and “Where” the object questions.

To the best of our knowledge, no research work highlights the use of the U-Net model in AVP, with an extensive comparison with other commonly used semantic segmentation models. Thus, the main contributions of this paper are:

1) Surveying the most recent research work on the two main methods used in the perception of autonomous vehicles: Semantic Segmentation and Object Detection.
2) Providing a comprehensive overview of the various deep learning method used in the AVP
3) Building five variants of the U-Net model
4) Building two variants of the SegNet, FCN-16, and FCN-8 models
5) Comparison between the above mentioned 11 models

The rest of this report is organized as follows: Chapter 2. is Literature Review and Related Work, Chapter 3. Discusses about the different U-Net models used, the adopted models to compare with the best performing U-Net model, the training of the models, and the performance evaluation metrics used to evaluate each model, Chapter 4. Discusses the obtained results and compares the models based on their results, and finally, Chapter 5. Is the conclusion and future directions.

Chapter 2: Literature Review and Related Work

Perception is the ability of an autonomous system to extract important information from the environment. It is a fundamental task to enable autonomous driving; it provides crucial information about the driving environment, including the accessible drivable areas, the locations, velocities, and prediction of the future state of the surrounding obstacles. Autonomous vehicles use LiDAR and Camera sensors for their perception to accurately detect obstacles and take the appropriate actions for a given scenario to avoid potential accidents. The essential tasks for a safe driving experience are Semantic Segmentation and Object Detection; these tasks are summarized in the following sections.

2.1 Semantic Segmentation

Autonomous vehicles rely heavily on semantic segmentation to navigate through routes. It operates by assigning each pixel in the image a particular class, and all the pixels that belong to a specific class are assigned a single color. As shown in Figure 1, vehicles are painted red, vegetation is painted green, buildings are painted grey, etc.

FIGURE 1. Semantic Segmentation on Cityscapes dataset

In [1], the authors tackled the problem of validating the performance of semantic segmentation algorithms under various operating conditions of autonomous vehicles, such as precipitation and illumination. Because even a slight variation in the environmental conditions could affect the classification performance and accuracy of the segmentation model, which can lead to catastrophic consequences. To solve this challenging problem, they proposed a pipeline that incorporated a Lidar sensor to test the performance of the semantic segmentation of a particular model in different real-world scenarios. They were able to distinguish the boundaries of the road around the vehicle. They automatically generated a large amount of ground truth road labels by testing the geometric properties of the surrounding Lidar points. They chose the 'Road' class from the semantic segmentation output to compare it with the ground truth generated by the Lidar sensor to prove the possibility of obtaining a measure of the classification performance and accuracy to validate the model. They also collected a weekly dataset of the area around their campus for 6 months to analyze the trained segmentation network performance and compare the validation accuracy of a model against datasets with different lighting and weather conditions. They used the proposed validation pipeline to compare the performance of two different semantic models, namely ENet and Bonnet. By performing these comparisons, they concluded that the best model selection depends on the operating conditions, and the accuracy of the models varies depending on the dataset. The authors in [2] tackled the problem that current semantic segmentation models face, which is the edge of the detected object is not clear. Their method utilized EfficientNet as the backbone network, coord convolution is applied to low features to add the position information, because of this addition the performance of this method was higher than the existing semantic segmentation models, the experiment showed that the application of Direction Convolution led to a more accurate edge detection compared to existing techniques. The proposed method was validated

on the ‘Cityscape’ dataset and resulted in a high performance, particularly on people and bicycles of different shapes. In [3], the authors tackled the need for a large computational resource for spatial-to-temporal approaches implemented in autonomous vehicles when tracking the various patterns of spatial positions for their motion. They proposed a temporal-to-spatial approach to cope with the vehicle’s speed in autonomous navigation by sampling a 1-pixel line at each frame in the video. The temporal connection of lines from consecutive frames makes a road profile image consisting of vehicles, road, lane mark, roadside, etc., and turning and stopping of ego-vehicle. This approach reduces the processing data to a fraction of video to catch up with the vehicle driving speed. They used RGB-F images (where F is a channel that describes features around the sampling line) of the road profile to perform semantic segmentation to retrieve individual regions and their spatial relations on the road. They tested their proposed method on naturalistic driving video, and the results were promising.

A comparison of some of the current research work in semantic segmentation based on the used algorithm, available datasets, and the current challenges is provided in Table 1.

TABLE 1 Semantic Segmentation Approaches

Paper	**Algorithm**	**Dataset**	**Problem**
[1]	Enet & Bonnet	Cityscapes & USYD	Validating the performance of semantic segmentation algorithms under a variety of operating conditions
[2]	Efficient Net	Cityscapes	The edge of the detected object is not clear
[3]	Road Profile Semantic Segmentation	Self-collected	The need for large computational resources for spatial-to-temporal approaches

2.2 Object Detection

Object detection is a fundamental task in any autonomous driving system, which identifies and locates object classes of interest in an image and creates a bounding box around those objects. Some of the famous object detectors include YOLOv2, YOLOv3, and Viola-Jones algorithm. Others use more sophisticated deep learning-based models. A real-time classification based on the Real AdaBoost algorithm is introduced in [4]. Lidar 3D point clouds are used to compute various features of road objects. The proposed classifier achieved over 90% accuracy in a 50-meter range. This algorithm can be used for autonomous driving because it classifies an object in 0.07×10^{-3} seconds. The authors in [5] have tackled the problem of unreliable and noisy 3D maps generated by LIDAR sensors for precise mapping and localization of Autonomous vehicles due to the existence of moving objects in the map, which leads to bad localization. Their proposed system takes 3D points from LIDAR, camera images, and GPS/INS information as input and outputs a vehicle-free 3D point cloud map. They used YOLOv2 Vehicle Detection Network (YVDN) to find the bounding boxes of the vehicles in an image and used K-Frames forward-backward bounding box tracking algorithm to find the missing bounding boxes. The 3D points that fall into the detected bounding boxes are then removed from the LIDAR frame. They registered each vehicle-free LIDAR scan to a global coordinate based on the GPS data to reconstruct a vehicle-free 3D point map. They validated their proposed method on the Oxford RobotCar Dataset and proved to generate a precise vehicle-free 3D point cloud map. In [6], the authors built a system to detect the surrounding vehicles and warn the driver of potential collisions. The proposed method consisted of two parts is implemented in a Robot Operating System (ROS). The first part uses the YOLOv2 algorithm for vehicle detection in an autonomous vehicle environment and is configured to detect

four different classes of vehicles: trucks, buses, vans, and cars. The second part uses two ROS nodes, the first node is used for distance assessment in the Carla simulator, and the second node is used for real-world distance assessment. The evaluation of the proposed method showed promising results. The authors [7] focused on object detection and tracking, an integral part of Advanced Driver Assistance Systems (ADAS). Object detection and tracking provide necessary information for collision avoidance, emergency braking, path planning, etc. The authors used two object detection algorithms: Viola-Jones and YOLOv3. The Viola-Jones algorithm was used to create nine object detectors classified under four groups: traffic light detector, pedestrian detector, traffic sign detector, and vehicle detector. Viola-Jones was compared with YOLOv3 based on their Precision, Recall, and processing speed. It was concluded that YOLOv3 achieved higher Precision and Recall and has a shorter processing time than Viola-Jones. They also used Median Flow tracking and Correlation tracking methods for object tracking. Median Flow tracking has a faster processing time, but both methods achieved similar results in terms of Multiple Object Tracking Accuracy (MOTA). They validated the proposed method on various datasets, such as German Traffic Sign Recognition Benchmark, INRIA Person, Udacity, and Cars datasets. Table 2 provides a comparison study between some related work in the literature of object detection.

TABLE 2 Object Detection Methods

Paper	**Algorithm**	**Dataset**	**Problem**
[4]	Real AdaBoost	Self-collected	Real-time object classification using Lidar
[5]	YOLOv2 Vehicle Detection Network	Oxford RobotCar	3D maps are noisy due to moving objects which leads to inaccurate localization of Autonomous Vehicles
[6]	YOLOv2 Vehicle Detection Network	Self-collected	Detect surrounding vehicles & warn the driver of potential collisions

[7]	Viola-Jones, YOLOv3, Median Flow, Correlation tracking	German Traffic Sign Recognition Benchmark, INRIA Person, Udacity, Cars	Object detection and tracking

2.3 Deep Learning for Autonomous Vehicle Perception

Deep learning is the backbone of every autonomous driving system, it is being used by object detection and classification algorithms (Supervised Learning) to detect and classify obstacles around the vehicle. It is also used for decision-making (Deep Reinforcement Learning) based on the observed data. Autonomous vehicles extensively use Convolutional Neural Networks (CNN), one of the most famous deep learning models. A CNN model consists of three main layers: A Convolutional Layer is used to extract features from the input image by convolving (dot product) the input image with a filter of size M x M, and it outputs a feature map. A Pooling Layer is often placed after the convolutional layer to reduce the size of the feature map, reducing the computational cost of the model. A Fully Connected layer consists of neurons along with weights and biases. It is used to connect each neuron to all the neurons in the previous and the next layer. It takes the flattened image as a vector as its input and outputs the classification results.

2.3.1 Deep Learning for Semantic Segmentation

The authors in [8] address the lack of research in the real-time RGB-D fusion semantic segmentation domain, despite accessible depth information. They proposed a real-time fusion semantic segmentation network named RFNet. The encoder part consists of two independent branches to extract the features of the input RGB and Depth images separately. They chose ResNet-18 as the backbone model to extract the features from the input images due to ResNet-18's residual structure and moderate depth. Its small operation footprint makes it compatible with real-time applications. After every layer of ResNet-18, the output features from the Depth branch are

fed to the RGB branch after the AFC module. The SPP produces feature maps with multiscale information by collecting the fused RGB-D features from both branches. Finally, they used up-sampling modules to restore the resolution of the produced feature maps with a direct connection from the RGB branch and skipping the Depth branch. They also used Multi-dataset training to incorporate small obstacle detection to enrich the recognizable classes, which will help detect the unforeseen hazards in real-world scenarios. They used the 'Cityscapes' and 'Lost and Found' datasets to test their model, outperforming previous state-of-the-art semantic segmentation models on the 'Cityscapes' dataset with high accuracy. The authors in [9] proposed an encoder-decoder-based deep CNN model in autonomous vehicle scenarios semantic segmentation. The proposed model architecture is based on the VGG16 model. The encoder part of the architecture like VGG16 consists of 13 convolutional layers, which have 3x3 filters. The convolutional stride and the spatial padding are fixed to 1 pixel after each convolutional layer. To decrease the size of feature maps, Max-pooling layers are used. They used residual learning by performing element-wise addition and shortcut connection to preserve the context and spatial information. On the other side, the decoder part has a similar structure as the encoder, but with only a few differences, such as the convolutional layers are replaced by de-convolutional layers and the Max-pooling layers by Up-sampling layers. They validated their proposed model on two popular benchmark datasets, namely, 'Cityscapes' and 'CamVid.' The experiments incorporated comparative analysis with popular networks such as ENet and SegNet, proving that their model outperformed both ENet and SegNet. In [10], They argue that the existing Semantic Segmentation methods partition the images into several semantically meaningful parts to classify each part into one of the pre-determined classes, ignoring the different importance levels of classes. For example, bicycles, other cars, and pedestrians are much important than the buildings or the sky in the scene when driving

autonomously, so they need to be segmented as accurately as possible to avoid catastrophic incidents. They proposed ‘Importance-Aware Loss’ IAL to tackle this problem, emphasizing the importance of critical objects in an autonomous driving scene. The IAL is designed based on a hierarchical structure, such that classes with different importance levels are located on a different level of the hierarchy. They also derived the forward and backward propagation of the IAL on four deep neural networks, namely, FCN, ENet, ERFNet, and SegNet. And tested these four networks on the ‘CamVid’ and ‘Cityscapes’ datasets, which obtained improved segmentation results on the pre-defined important classes. Road lane marking and road edge detection on Lidar-based autonomous cars are addressed in [11].This includes the capability of obstacle avoidance but cannot detect road lane markings. They solved this problem by installing and calibrating a low-cost monocular camera on a Formula-SAE electric car equipped with a Lidar sensor. They first tested the system on video recording of local roads to ensure the feasibility of SegNet semantic segmentation. Then they tested on the Formula-SAE car with Lidar readings. The obtained results from the semantic segmentation performed on the CamVid dataset proved that lane markings and road edges could be classified using the proposed method. In [12], the problem of accurate road marking extraction is discussed. Addressing the complexity of road marking, they used a Dense Feature Pyramid Network (DFPN) based deep learning model, which concatenates the deep feature channels with shallow feature channels to help the shallow feature maps with abundant image details and high resolution utilize the in-depth features. The proposed deep learning model was trained end-to-end on mobile laser scanning (MLS) point cloud to extract the road markings. They optimized the deep learning model using the focal loss function. Extensive experiments had proved the proposed method outperformed the existing state-of-the-art methods in instance segmentation of road markings. In [13], a 3D Semantic Segmentation of point clouds in urban areas using deep

learning is introduced. They conducted a comparative study on three novel deep learning-based semantic segmentation algorithms, PointCNN, PointNet, and SPGraph. The algorithms were trained on an outdoor aerial survey point cloud dataset and were evaluated based on the overall accuracy. The evaluation showed that SPGraph, PointNet, and PointCNN achieved 83.4%, 83%, and 72.7% overall accuracy for 3D semantic segmentation.

2.3.2 Deep Learning for Object Detection

In [14], the authors proposed a method for object detection and identification. They utilized 3-D Lidar data to generate object region proposals. Then, they mapped those candidates onto the image space from which the ROI (Region of Interest) of the proposals are selected and input to a CNN model based on the VGG16 model to perform object recognition. Then, they combined the features of the last three layers of the CNN to extract multiscale features from the Region of Interests to precisely identify the sizes of every object in the scene. They evaluated the proposed model on the KITTI dataset and reached the following conclusions:

- The processing time of each frame is 66.79ms, which is suitable for real-time processing.
- 3-D Lidar produces 86 candidate object-region proposals, compared to a sliding window that produces thousands of candidates per frame.
- The average identification accuracy of pedestrians and cars is 78.18% and 89.04%, respectively.

In [15], the authors designed a real-time pedestrian detection system for autonomous vehicles using CNN. They created the system from scratch without using any available libraries. They evaluated their model on three datasets: INRIA, PETA-CUHK, and real-time video input and achieved accuracy ranging between 96.73% and 100%. Deploying advanced Deep Convolutional Neural Network (DCNN) detectors in autonomous vehicles with limited memory and computing power is

a challenging task [16]. To solve this problem, it is necessary to design lightweight and robust detectors. Recently, a novel algorithm has been proposed named ‘Group Convolution’ to make the detection network faster and lighter by reducing the floating-point operations. But the existing guidelines do not indicate the optimal number of groups in the Group Convolution to maximize the detection speed. This paper introduced three new guidelines to indicate the optimum number of groups needed to design a fast and lightweight detector and named this detection network ‘DenseLightNet’. The proposed method runs three times faster than the existing state-of-the-art detector YoloV3 and weights 10.1MB compared to the YoloV3’s 247MB. A Deep Neural Network (DNN) based object detector called Single-Shot Detector (SSD) is designed in [17]. The SSD architecture consists of a base network and an auxiliary network. VGGNet is used as a base network for good quality classification, and the auxiliary network is used to predict detection at multiple feature maps. A non-maximum suppression follows the base network and the auxiliary network to decide the final detections. The proposed method was evaluated on the KITTI dataset, and it outperformed the original object detection model based on precision by 6%.

In [18], a method for simultaneous detection of people, vehicles, lanes, and non-motor vehicles using RGB-D images is discussed. The task consists of two parts: the detection of vehicles, people, and non-motor vehicles as a general detection task, and lane detection as a segmentation task. They used two separate networks to improve the accuracy and speed, the first network is called LaneNet to segment the lanes, and the second is Faster-RCNN to detect the rest. For separate training and simultaneous detection of both networks, they introduced a real-time synchronization method with multi-GPU. The detection frame rate of the system reached 15 FPS with four 1080Ti GPUs. The system was evaluated on a self-collected dataset, and it achieved high accuracy. They also tested the system in a real-time scenario on the streets of China, which proved that the system could be

applied in real-time autonomous driving. In [19], the authors address the two main tasks involved in tracking and localizing vehicles and objects surrounding an autonomous vehicle: detecting and classifying obstacles. They proposed a region-based convolutional neural network named Faster-RCNN trained with PASCAL VOC dataset to detect and classify obstacles such as pedestrians, vehicles, animals, etc. This method was implemented on a Titan X GPU and achieved a detection frame rate of 10 FPS on a VGA resolution image frame. The achieved fast frame processing rate ensures the usability of this system on highways. They validated the detection and classification performance of the system on the KITTI and iRoads datasets. They concluded that the performance did not vary on different shapes, views of an object, and different climate and lighting conditions. In [20], a model to predict the future trajectory of the objects using the Gated Recurrent Unit (GRU) is introduced. This model understands the behavior of the surroundings in a mixed scene of bicycles, vehicles, and pedestrians. Since these objects have different behaviors, they applied different models to other categories. The proposed method takes three observed trajectories with varying time steps as input and predicts an accurate future trajectory. The model was then compared with GRU and LSTM and resulted in a minor Mean Absolute Error (MAE) and converged faster than GRU and LSTM. Deep Reinforcement Learning-based for obstacle detection and autonomous navigation, named Deep Q Network (DQN,) on a simulated car in an urban environment, has received widespread attention in the last few decades [21]. The model takes input camera and laser sensor data placed on the car's front end. They also designed a prototype of a cost-efficient high-speed car to run the algorithm in real-time. They placed a Hokuyo Lidar sensor and a camera on the car and used an Nvidia-TX2 GPU to run the deep learning models.

In [22], the brake-lights recognition problem is presented with a focus on deep learning. The “Brake Lights Patterns” (BLP) are learned using a Multi-Layer Perceptron (MLP) based classifier

that classifies the vehicles in an image as “Normal” or “Brake”. The authors explored road segmentation and novel vanishing point ROI determination methods to speed up the detection and improve the system's robustness. The validation results conducted on on-road videos collected by the authors have shown the efficiency and robustness of the proposed method. In [23], the authors worked on autonomous vehicle learning simulation results to drive in a simple environment containing static obstacles and lane markings. The algorithm takes an image of the street captured by the car front camera as an input. It computes the Q values representing the rewards that correspond to future actions taken by the autonomous vehicle. The actions are angles through which the vehicles steer at a fixed speed. The system enforces the car to act with the highest reward (Q value). The simulation results showed a high accuracy achieved by the model by following the lanes and avoiding obstacles. Vehicle speed control using Reinforcement Learning methods is addressed in [24]. Their main motivation was the instability of the Q-learning algorithm in some games in the Atari 2600. They used an algorithm called Double Q-learning to control the vehicle's speed based on the surrounding environment. They proposed a new method that depends on the direct perception approach called the integrated perception approach to construct the environment. Both low dimensional data processed from the sensors and high dimensional data with road information from the video make up the input of the Double Q-learning model. Experimental results have shown that the Double Q-learning algorithm outperformed the traditional Q-learning algorithm regarding policy quality and value accuracy. The total model score is 271.73% times that of Q-learning.

In [25], a comparative study on object recognition using deep convolutional neural networks (CNN) in autonomous vehicle environments is presented. They used four well-known CNN models, Faster R-CNN Inception V2, Faster R-CNN Resnet 50, SSD Inception V2, AND Faster R-

CNN Resnet 101. These models were pre-trained on the COCO dataset, and they were retrained with the new dataset using transfer learning. The new dataset was formed using GRAZ-01 and GRAZ-02 datasets and consisted of 517 images of 10 objects: Cars, Bicycles, Pedestrians, and 7 traffic signs. The experimental results have shown that Faster R-CNN outperformed the model models, with an accuracy of 85.1%. A collision avoidance system for autonomous vehicles based on Reinforcement Learning can learn from mistakes and readdress its movement accuracy [26]. They used the Q-learning method to record and update the Q-values in a table for different movements, which will be used by the autonomous vehicle to determine how and where to move.

A deep neural network was used to learn the Q-value table, which encounters many situations from different actions performed by the autonomous vehicle. The input to the model is 10000 images captured by a depth camera placed on the car's front end. The model was trained for 9000 epochs and achieved an obstacle avoidance rate of 95%. The autonomous braking problem is analyzed and discussed in [27] through precise decision-making and control to reduce accidents. They proposed a Deep Reinforcement Learning-based autonomous braking system in emergencies. They considered three key influencing factors: accuracy, efficiency, and passengers' comfort. These factors were fully satisfied by the proposed system. They designed a multi-objective reward function for compromising the passengers' comfort, the degree of the accident, and the achieved rewards of different brake moments. To solve the autonomous braking problem, they adopted an actor-critic (AC) algorithm called Deep Deterministic Policy Gradient (DDPG), which improves the system's efficiency and makes it stable in continuous control tasks. They evaluated the proposed method through extensive simulations, which proved its efficiency in driving safety, decision-making accuracy, and learning effectiveness.

A deep learning model for 3D object proposal generation and detection from point cloud data called PointRCNN is proposed [28]. The framework is composed of two stages: The first stage generates a small number of high-quality 3D proposals in a bottom-up manner by segmenting the point cloud data into background and foreground points, unlike previous methods that used to generate proposals by projecting point cloud to bird's view or from RGB images. The second stage transforms the segmented points in the first stage to canonical coordinates to learn much better local spatial features. Those spatial features are combined with global semantic features for accurate confidence prediction and box refinement. The experiments performed on the KITTI dataset showed that the proposed PointRCNN architecture outperforms state-of-the-art methods by only using point cloud as its input data.

For self-driving, a deep learning system can use LiDAR point clouds and depth image-based rendering (DIBR) for self-driving [29]. The DIBR is used to generate parallax map information and obtain the depth image, which is then combined with LiDAR point cloud to repair the objects in the point cloud image. They also combined the Histogram Equalization and Optimal Profile Compression (HEOPC) with the accuracy of deep learning to optimize the color image enhancement. Based on the restored point cloud image, they used a cutting algorithm to divide the areas of interest, such as cars, people, and bus and train a MobileNet-YOLO model to identify those three objects. Detecting 3D objects in point clouds is challenging [30]. This problem was previously solved by projecting a 3D point cloud into 2D images. This means transforming the 3D detection problem into 2D detection. This method produces multiple 2D detection tasks, which increases the complexity and limits the performance of the 2D detection algorithm. To solve this problem, the authors proposed using a Convolutional Neural Network (CNN) model to perform the 2D detection task because CNN can predict multiple classes of objects using the same network

without using an individual detector for each category. They concatenated two early rejection networks with binary outputs before the detection network to improve the detection efficiency. Extensive experiments have shown that the proposed method achieved a competitive performance, with at least ten times the speed of the latest 3D point cloud detection methods.

In Table 3, a comprehensive comparative study is provided among the state-of-the-art deep learning methods in semantic segmentation and object detection.

TABLE 3 Comparison Between Deep Learning Models

Paper	**Used Algorithm**	**Dataset**	**Problem**
[8]	RFNet	Cityscapes & Lost and Found	Lack of real-time RGB-D fusion semantic segmentation work
[9]	VGG16 & Residual Encoder-Decoder	Cityscapes & CamVid	Residual Encoder-Decoder Network for Semantic Segmentation
[10]	FCN, SegNet, Enet, ERFNet	Cityscapes & CamVid	Semantic Segmentation methods give the same importance to all classes
[11]	SegNet	CamVid	Lidar-based autonomous vehicles are unable to detect road markings and road edges
[12]	DFPN	Self-collected	Road Marking Instance Segmentation Using MLS Point Clouds
[13]	PointNet, PointCNN, SPGraph	Fused 3D point cloud	3D Semantic Segmentation of Large-Scale Point-Clouds in Urban Areas Using Deep Learning
[14]	VGG16	KITTI	Object detection and identification using 3-D Lidar
[15]	CNN	INRIA, PETA-CUHK	Pedestrian detection using CNN programmed from scratch
[16]	DenseLightNet	City, Pascal VOC	Limited computing power and memory on Autonomous Vehicles for advanced DCNN
[17]	Single-Shot Detector	KITTI	On-road object detection using DNN
[18]	Faster-RCNN, LaneNet	Self-collected	RGB-D based real-time multiple object detection and ranging system
[19]	Faster-RCNN	KITTI, iRoads	On-road obstacle detection and classification

			using deep learning to track in a high-speed AV environment
[20]	GRU, LSTM	KITTI	The trajectory of Prediction of Immediate Surroundings Using Hierarchical Deep Learning Model
[21]	Deep Q Network	Simulated the model	Deep Reinforcement Learning for obstacle avoidance and autonomous navigation
[22]	CNN (AlexNet)	Self-collected	Appearance-based Brake-Lights recognition using deep learning
[23]	Deep Q Network	Simulated the model	Deep Reinforcement Learning for obstacle avoidance and lane detection
[24]	Double Q-Learning	Simulated the model	Instability of the Q-learning algorithm in speed control of vehicles in some games in the Atari 2600
[25]	Faster R-CNN	GRAZ-01, GRAZ-02	A comparative study on different CNN based object detection models
[26]	Q-Learning	Self-collected	Reinforcement Learning based collision avoidance system
[27]	Deep Deterministic Policy Gradient (DDPG)	Simulated the model	Deep reinforcement Learning-based autonomous braking decision-making strategy in an emergency
[28]	PointRCNN	KITTI	3D Object Proposal Generation and Detection From Point Cloud
[29]	MobileNet-YOLO	KITTI	Self-driving Deep Learning System based on Depth Image Based Rendering and LiDAR Point Cloud
[30]	CNN	UWA 3D Object, CMU Oakland 3-D Point Cloud, Washington Urban Scenes 3D Point Cloud	3D point cloud object detection with multi-view convolutional neural network

Chapter 3: LEVERAGING U-Net for URBAN SCENE SEGMENTATION

To perform semantic segmentation for scene understanding in autonomous vehicles, we have implemented five different variations of the U-Net model. The U-Net model was previously designed and implemented exclusively for medical image segmentation tasks. As the name of the model may imply, the model architecture has the shape of the letter 'U', as shown in Figure 2.

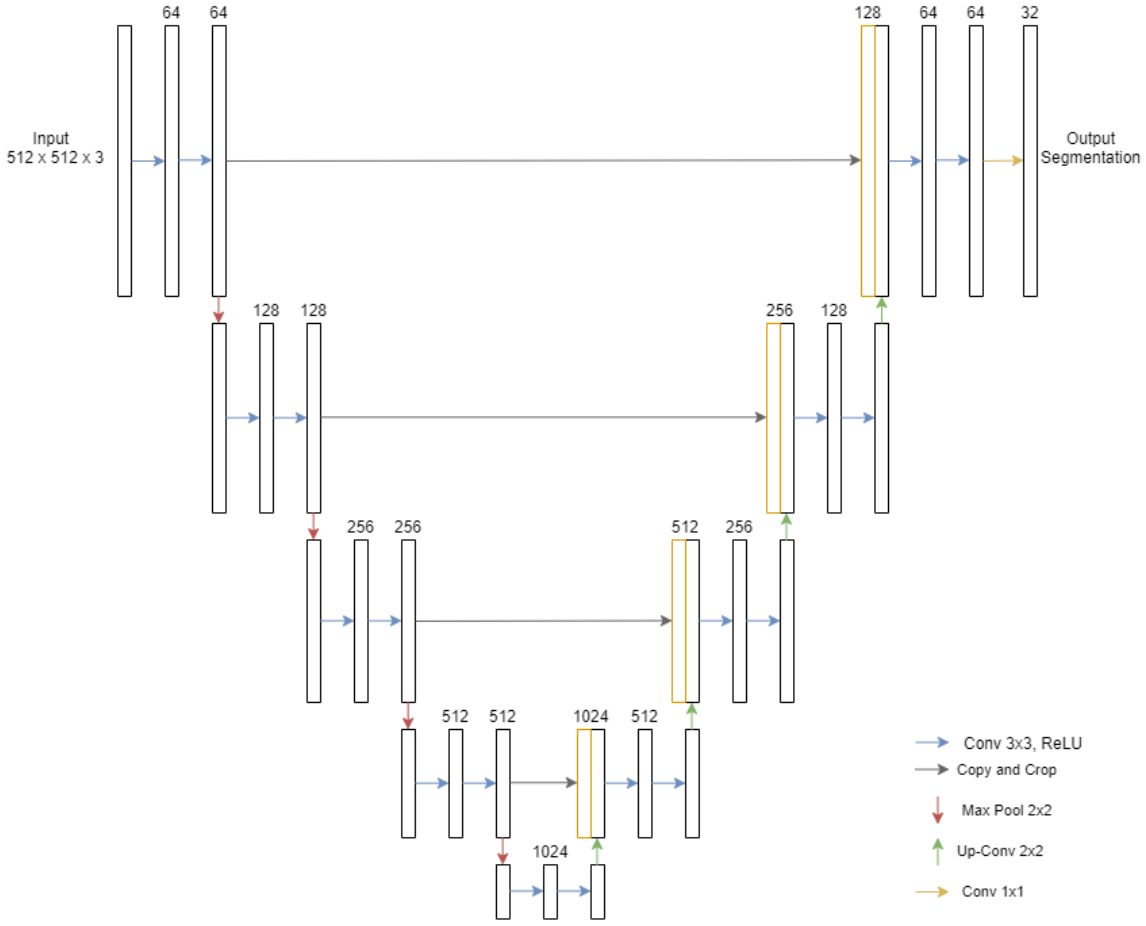


FIGURE 2 The U-Net Architecture

The U-Net consists of two paths, a contracting path and an expansive path. The contracting path also called the down-sampling path, consists of repeated two 3 x 3 convolutions, with a Rectified Linear Unit (ReLU) as their activation function, followed by a 2 x 2 max pooling operation with a stride of 2 used to down-sample. Each down-sampling step in the contracting path, the number

of feature channels is doubled, the image size path gradually decreases the depth increases. The expansive path, also called the up-sampling path, consists of up-sampling of feature map and a 2 x 2 convolution to halve the number of feature channels, ana concatenation with the corresponding parallel cropped feature map on the contracting path, two 3 x 3 convolutions, with Rectified Linear Unit (ReLU) as their activation function. The final layer is a 1 x 1 convolutional to map the feature vectors to the corresponding number of classes. The size of the image in the expansive path gradually increases, and the depth decreases. We trained two other U-Net models with four times smaller feature channels, as shown in Figure 3, one with ReLU as its activation function and the second model with LeakyReLU as its activation function.

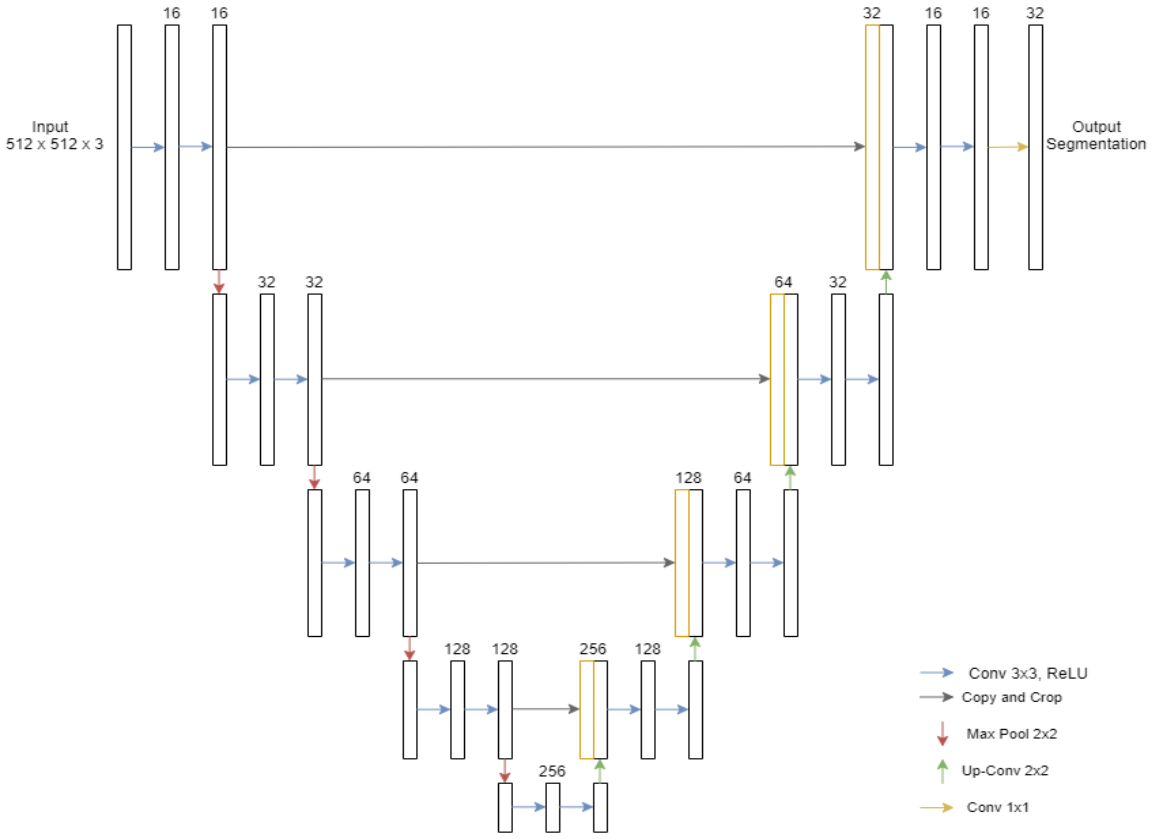


FIGURE 3 The U-Net Architecture with Smaller Feature Channels (Small U-Net)

We also trained two more U-Net variants and called them “Long U-Net”, because we added two layers on the contracting path and two layers on the corresponding expansive path, as shown in Figure 4. To help the model generalize better, we used a regularization technique called Dropout. We trained one “Long U-Net” model with a Dropout rate of 0.5 and trained another “Long U-Net” with a Dropout rate of 0.7 to analyze the model performance.

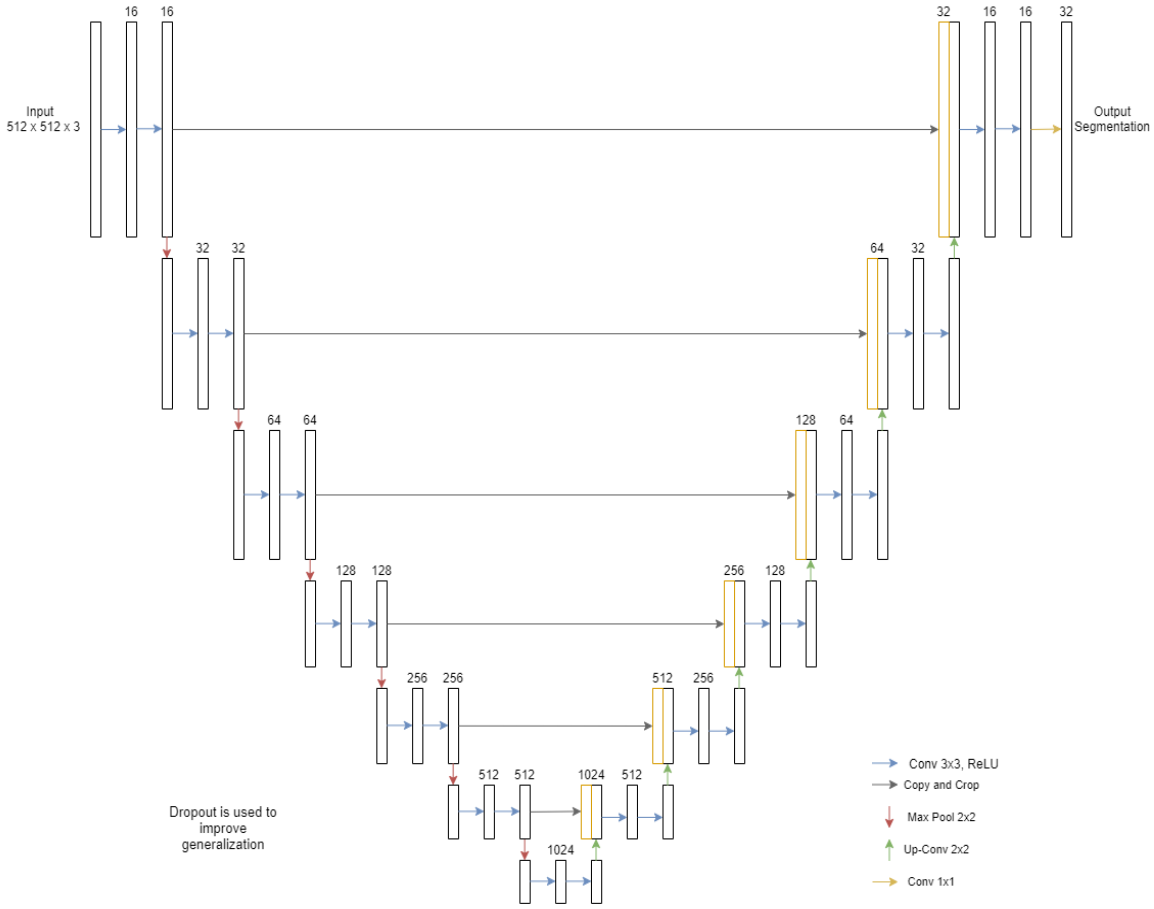


FIGURE 4 The Long U-Net Architecture

3.1 The Adopted Models

We have adopted three commonly used semantic segmentation models: SegNet, FCN-16, and FCN-8, to compare their performance with the best performing U-Net model. Other than the original three models, we have built three other different with the only difference is the Dropout

technique added to each model because the best performing U-Net is the one with a Dropout rate of 0.5.

3.1.1 The SegNet Model

The SegNet model consists of an encoder and a corresponding decoder network, and at the final layer, it performs pixel-wise classification of the input image, as shown in Figure 5. Inspired by the VGG-16 network, designed for object classification, they used 13 convolutional layers in the encoder network. Still, they discarded the fully connected layers to retain higher resolution feature maps at the encoder output. By discarding the three fully connected layers of VGG-16, the authors drastically reduced the number of SegNet model parameters. Each encoder layer has a corresponding decoder layer, meaning the decoder network also has 13 layers. The decoder output is fed to a soft-max classifier which produces class probabilities for each pixel, and the prediction corresponds to the class with maximum probability at every pixel.

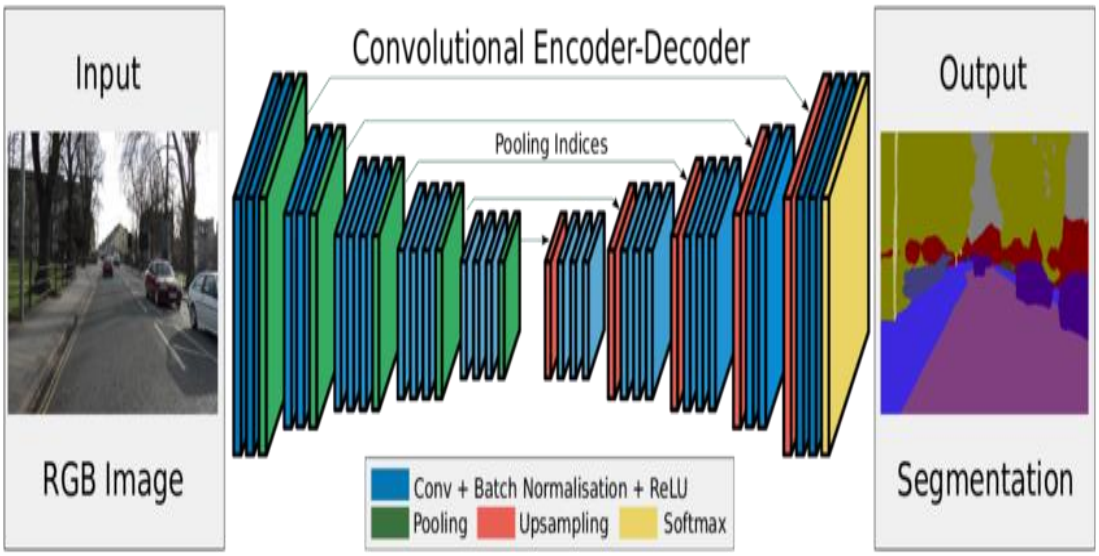


FIGURE 5 The SegNet Architecture [31]

3.1.2 Fully Convolutional Network (FCN-16, FCN-8)

We have implemented the FCN-16 and FCN-8 only because FCN-32 had proven its poor performance in the literature, because at the output of conv7, as shown in Figure 6 below, the image size becomes very small, to make the segmentation output have the same size as the input image 32 x up-sampling is performed, which makes the output very rough because when going deeper the spatial location information is lost. That is why FCN-16 and FCN-8 perform better because they both use two and four times less up-sampling. In the FCN-16 network, the output of conv7 is 2 x up-sampled and fused with pool4 and performed 16 x up-sampling. In the FCN-8 architecture, the output of conv7 is 4 x up-sampled and fused with 2 x pool4 and pool3, then performed 8 x up-sampling.

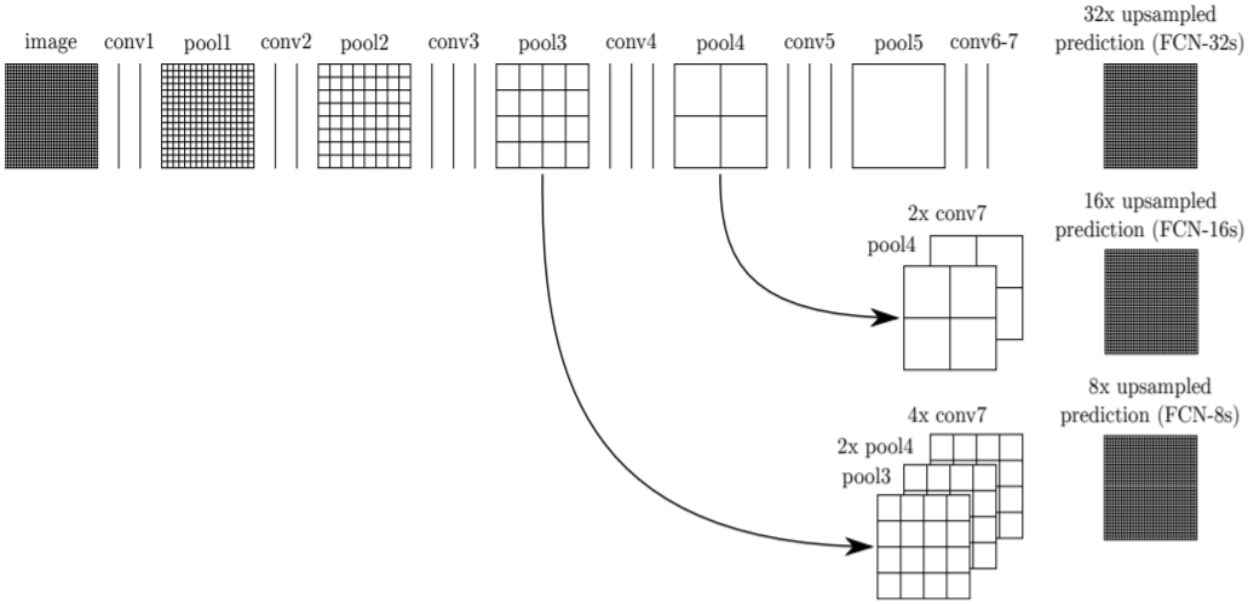


FIGURE 6 The Architecture of FCN (FCN-32, FCN-16, FCN-8) [32]

Training

The code was written in the Python3 programming language, using the Tensorflow library as the backend and the Keras library as the frontend. The models were trained on the Google Colab platform using the provided GPU by Google. We used "Adam" as the optimizer for all models, the "Categorical Cross

Entropy" as the loss function, and the batch size was set to 32, as per the best practice in Machine Learning research. The maximum number of epochs (iterations) was set to 100. Still, we used the early stopping method to stop training when the models' validation mean Intersection over Union (mIoU) does not improve after ten epochs.

Due to memory and GPU time restrictions imposed by Google Colab, we had to use a widely used dataset with a small number of images, called the Cambridge – Driving Labeled Video Database (CamVid) [33], which consists of 701 overall images.

3.2 Performance Evaluation Metrics

Performance evaluation is required to evaluate and optimize any machine learning model and compare it with other models. Different evaluation metrics are used in the literature; this section describes the most efficient and widely used metrics in semantic segmentation tasks. Intersection Over Union (IoU) matric, also known as Jaccard Index, is widely used to evaluate semantic segmentation models. It computes the percent overlap between the ground truth mask and the prediction output. As shown in Eq.1, IoU measures the number of common pixels between the prediction and ground truth masks and divides it by the total number of pixels present in both masks. Multi-class segmentation tasks use the mean Intersection Over Union (mIoU) metric for model evaluation, which first computes the IoU of each class and then computes the average overall classes.

$$IoU = \frac{Target \cap Predicted}{Target \cup Predicted} \qquad (1)$$

Accuracy is the most commonly used evaluation metric in Machine Learning research, but it is unreliable in semantic segmentation tasks. It measures all the correctly identified classes and is helpful when all the classes are equally important. The Accuracy is calculated by Eq.2.

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + True\ Negative + False\ Negative} \quad (2)$$

We used F1-Score, a better evaluation metric than Accuracy for imbalanced class distribution, and it is measured by calculating the harmonic mean of the Precision and Recall. The F1-Score is calculated by Eq.5.

$$Precision\ (P) = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (3)$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (4)$$

$$F1 - Score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \quad (5)$$

We also used the Dice Coefficient, which is similar to IoU, because they are positively correlated. It calculates the area of Overlap between the Ground Truth and the Predicted mask and divides it with the total number of pixels in both masks, and multiplies the result by 2. The Dice Coefficient is calculated by Eq.6.

$$Dice\ Coefficient = 2 * \frac{Area\ of\ Overlap}{Total\ Number\ of\ Pixels\ in\ both\ Masks} \quad (6)$$

Chapter 4: Experimental Results and Analysis

In this chapter, we will showcase and analyze the results of implementing eleven models, and will compare them based on their Accuracy, Loss, mIoU, F1-Score, and Dice coefficient. And we will also demonstrate the performance of each model on three different images from the test set to be able to visually prove the numerical results we obtained.

4.1 Dataset

The dataset we used is called The Cambridge-Driving Labeled Video Database (CamVid), it provides per-pixel semantic segmentation of over 700 images and their corresponding Ground Truth masks (labels), 367 training, 101 validation, and 233 test pairs of 32 semantic classes. The semantic classes are of the commonly existing objects in a regular driving scene, ranging from Cars, Pedestrians, Animals, Buildings, sidewalks, Traffic Lights, and many more. The overall database consists of ten minutes of high quality 30HZ footage, and the images we used to train our models, were captured at 1HZ.

4.2 Preprocessing

The images and masks are in separate folders, we paired each image with its corresponding mask, both images and masks are resized to 512 x 512. Images are converted to numpy arrays for easier tensor calculations and normalized by dividing by 255. The masks are also converted to numpy arrays and are mapped to the corresponding classes, the classes are given in an excel sheet with the ‘r’, ‘g’, and ‘b’ values of each class.

4.3 Experimental Results

After building a total of eleven models: 5 U-Net, 2 SegNet, 2 FCN-16, and 2 FCN-8, we can conclude that the Long U-Net with Dropout=0.5 performed the best, based on the mean Intersection over Union (mIoU) evaluation metric, with mIoU of 0.5731, as shown in Table 4, and Figure 12. The superiority of this model lies in the depth of its architecture. The Dropout regularization was used, which improved the generalization of the model, and because the model

converged at the 100th epoch, unlike other models that converged at a much smaller epoch. We also compared the models in terms of their Loss, Accuracy, F1-Score, and Dice coefficient. The U-Net model with ReLU activation function recorded the lowest loss of 0.3823. The Long U-Net with a Dropout rate of 0.7 recorded the highest accuracy of 89.85% and the highest Dice Coefficient 0.919, the Long U-Net with a Dropout rate of 0.5 recorded the highest F1-Score of 0.6384, as shown in Table 4.

TABLE 4 A Comparison Between the Models

Model	**Loss**	**Accuracy**	**mIoU**	**F1-Score**	**Dice Coef.**
U-Net ReLU	**0.3823**	89.74%	0.5366	0.6045	0.9183
Small U-Net ReLU	0.5254	87.96%	0.5331	0.5954	0.9029
Small U-Net Leaky ReLU	0.4436	87.80%	0.5203	0.5767	0.9039
Long U-Net(Dropout=0.7)	0.3971	**89.85%**	0.5442	0.6098	**0.9190**
Long U-Net(Dropout=0.5)	0.3901	89.13%	**0.5731**	**0.6384**	0.9139
SegNet	0.4978	85.37%	0.4861	0.5373	0.8820
SegNet (Dropout=50)	0.5652	83.97%	0.4652	0.5031	0.8665
FCN-16	0.5027	84.92%	0.5000	0.5623	0.8805
FCN-16 (Dropout=50)	0.4482	85.85%	0.5102	0.5639	0.8834
FCN-8	0.4860	86.03%	0.5039	0.5686	0.8891
FCN-8 (Dropout=50)	0.4216	87%	0.5176	0.5744	0.8938

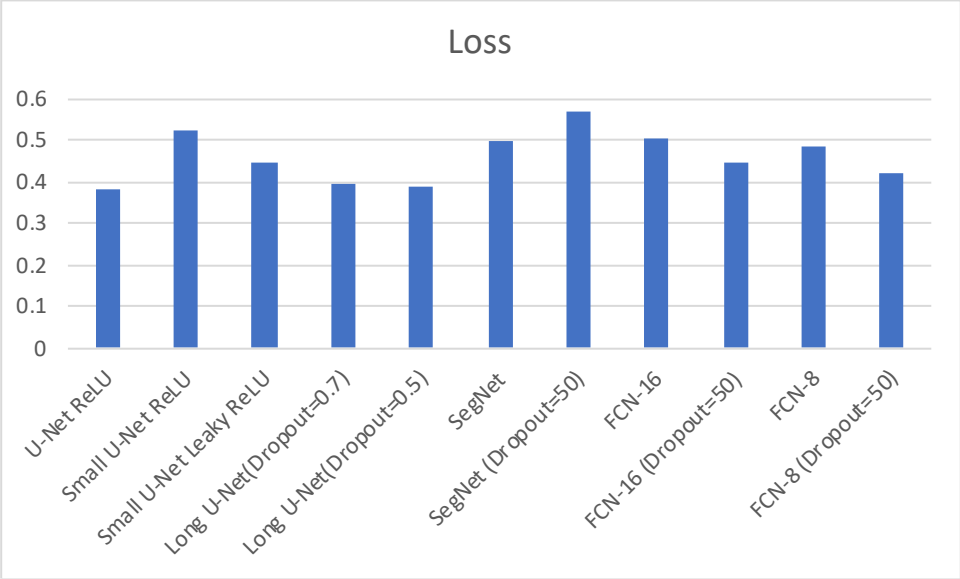


FIGURE 7 The Comparison in Terms of Loss

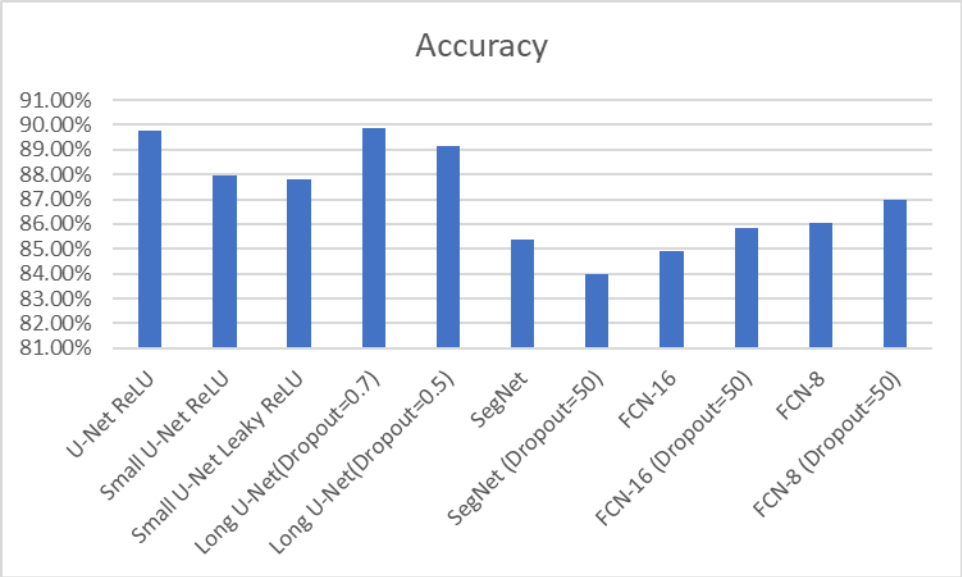


FIGURE 8 The Comparison in Terms of Accuracy

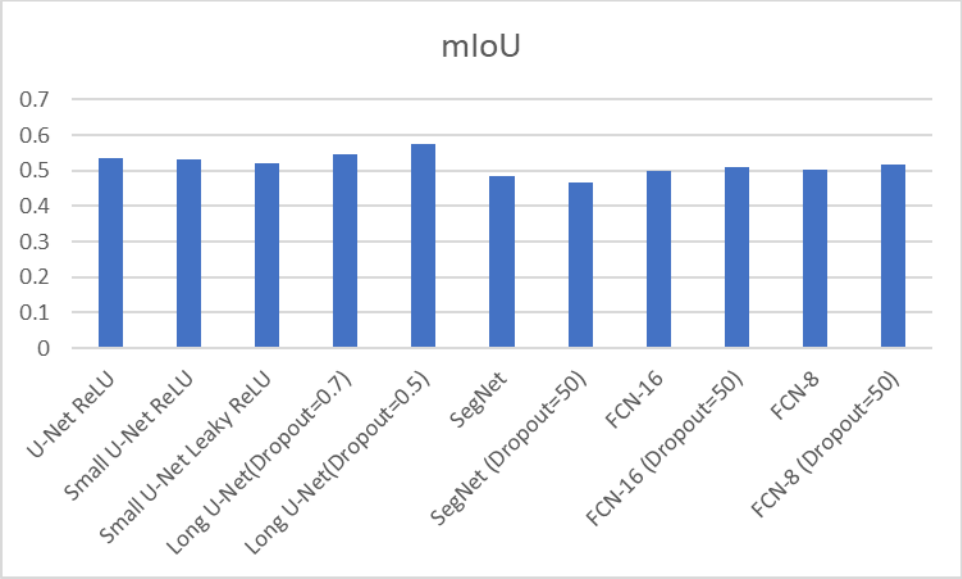


FIGURE 9 The Comparison in Terms of mIoU

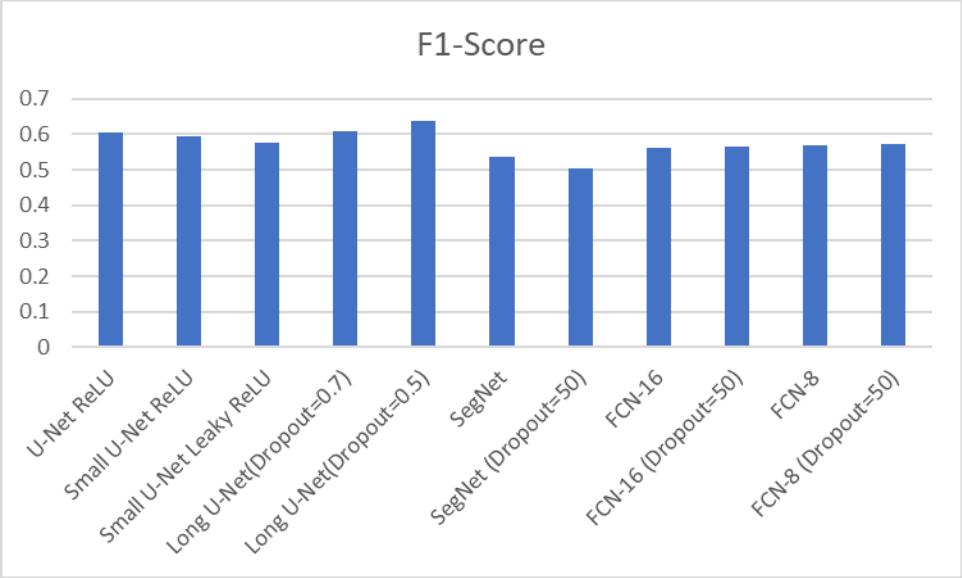


FIGURE 10 The Comparison in Terms of F1-Score

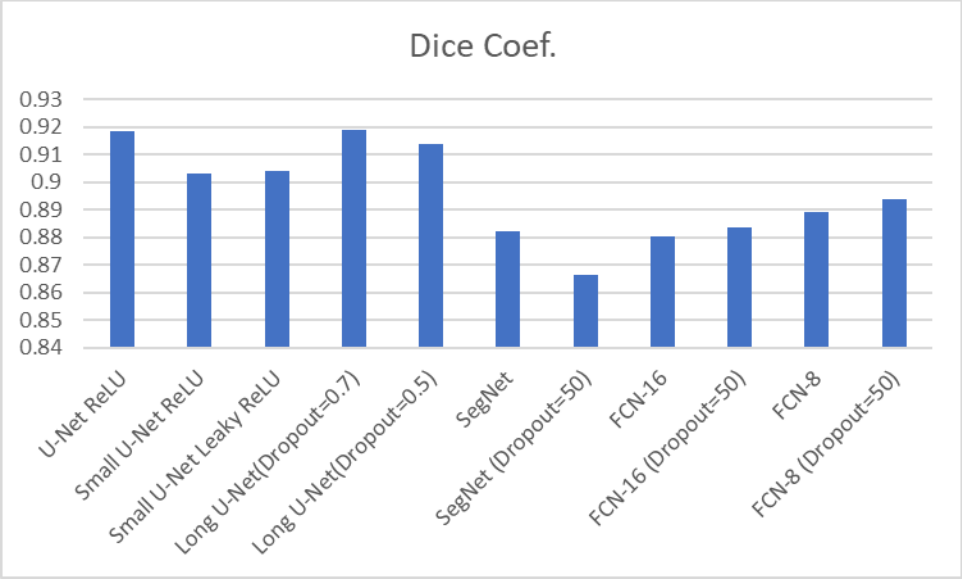


FIGURE 11 The Comparison in Terms of Dice Coefficient

Below are the mIoU and the Loss graphs of the best performing variant of each of the four models. Figures 12 and 13 are the mIoU and Loss graphs of the best performing U-Net model, Figures 14 and 15 are the mIoU and Loss graphs of the best performing SegNet model, Figures 16 and 17 are the mIoU and Loss of the best performing FCN-16 model, and finally the Figures 18 and 19 are the mIoU and Loss graphs of the best performing FCN-8 model.

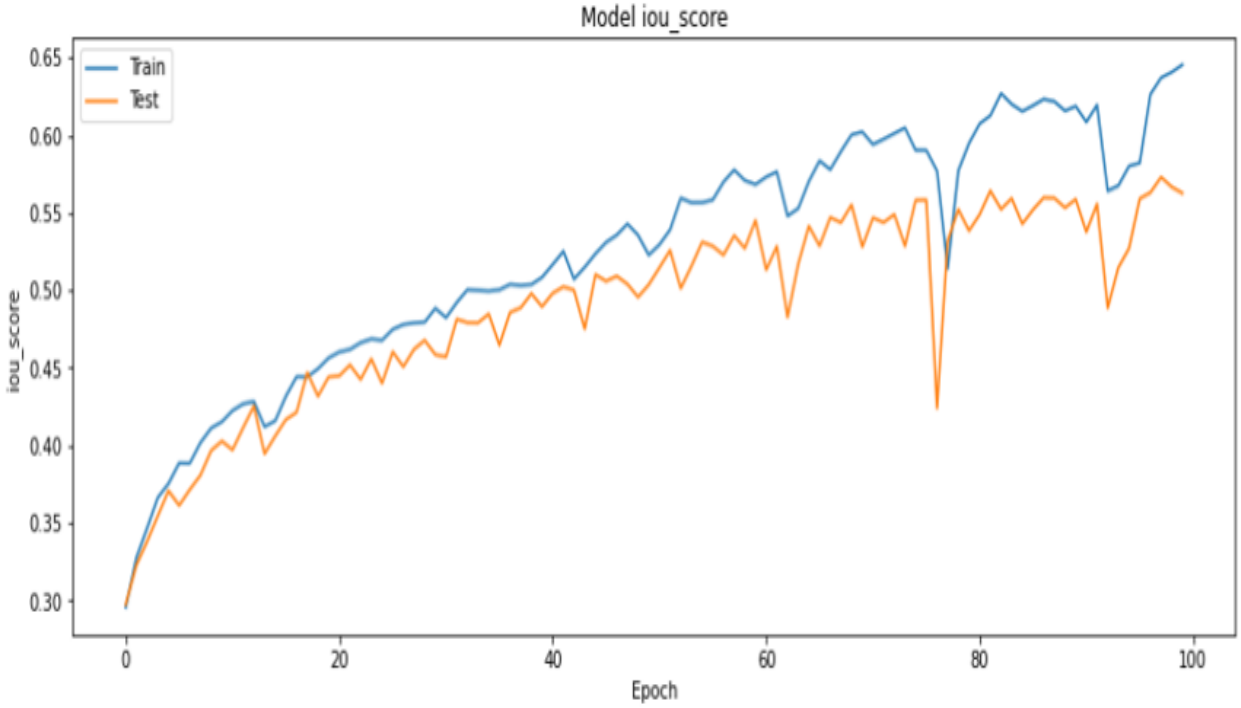


FIGURE 12 The mIoU of the Long U-Net Model with Dropout=0.5

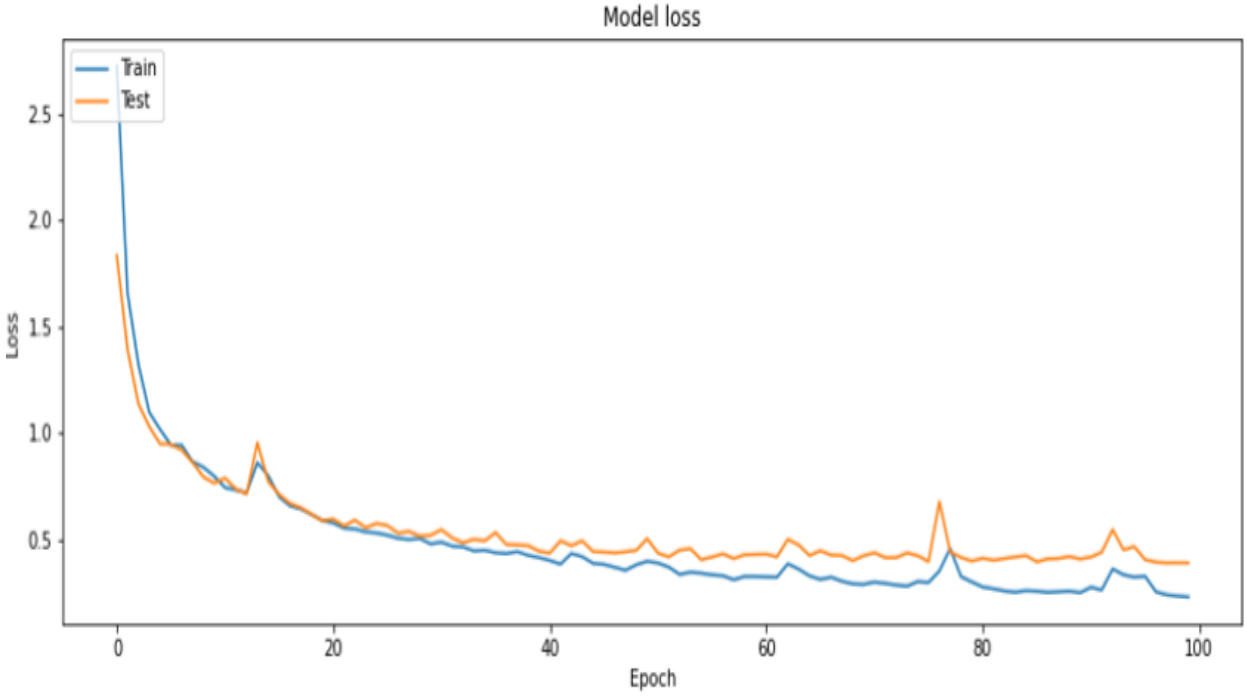


FIGURE 13 The Loss of the Long U-Net Model with Dropout=0.5

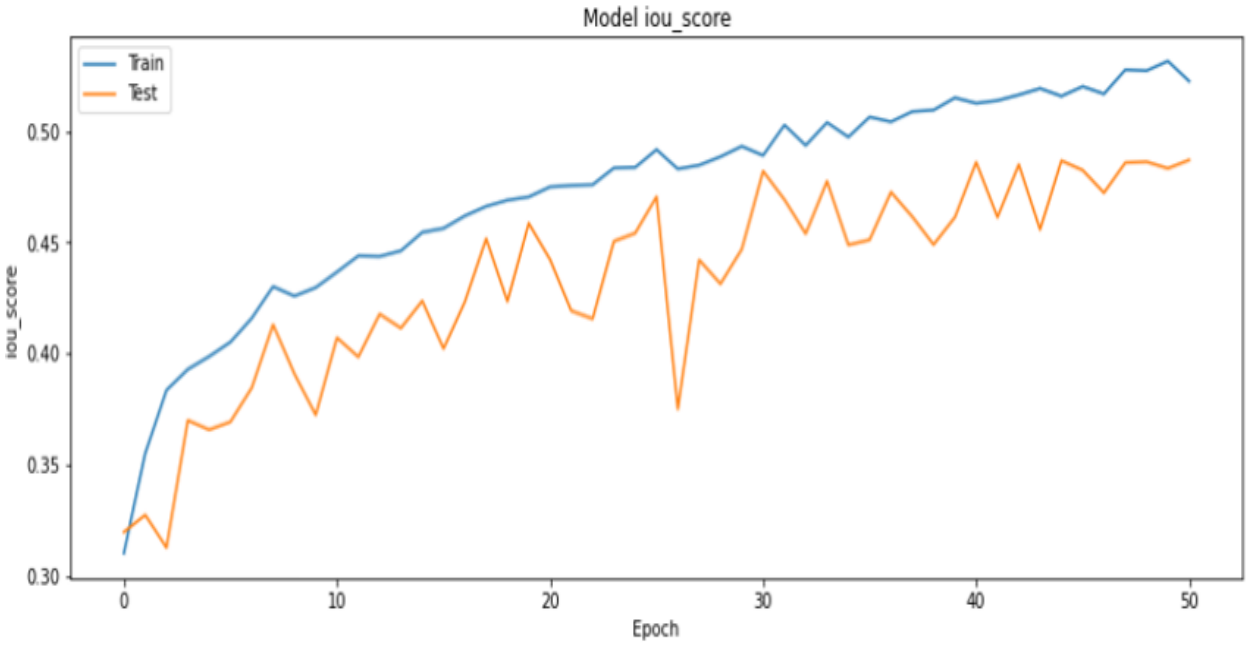


FIGURE 14 The mIoU of the SegNet Model

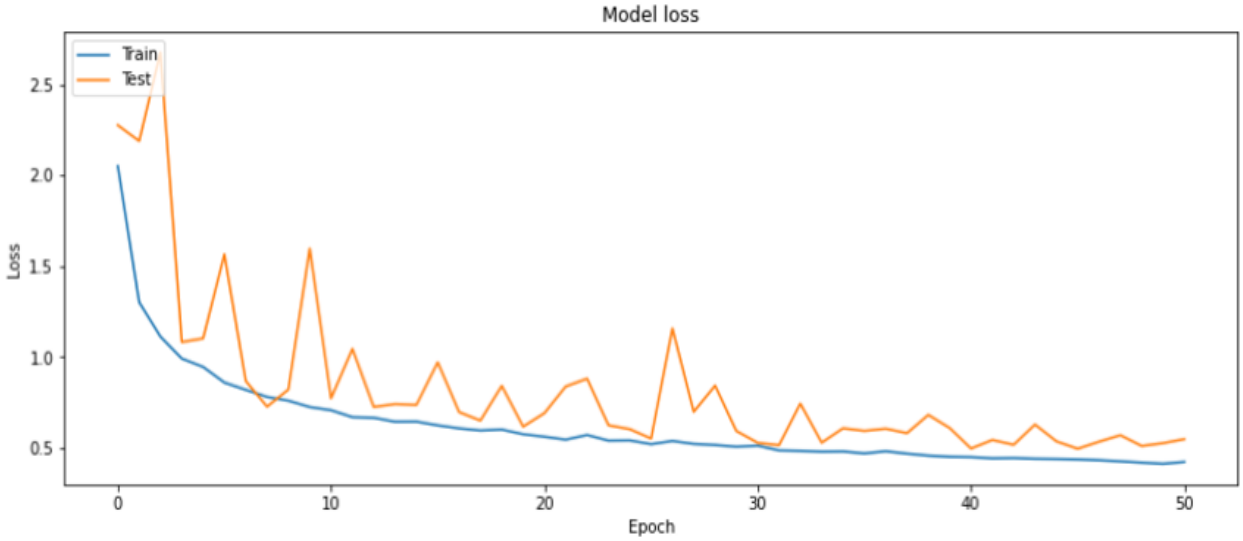


FIGURE 15 The Loss of the SegNet Model

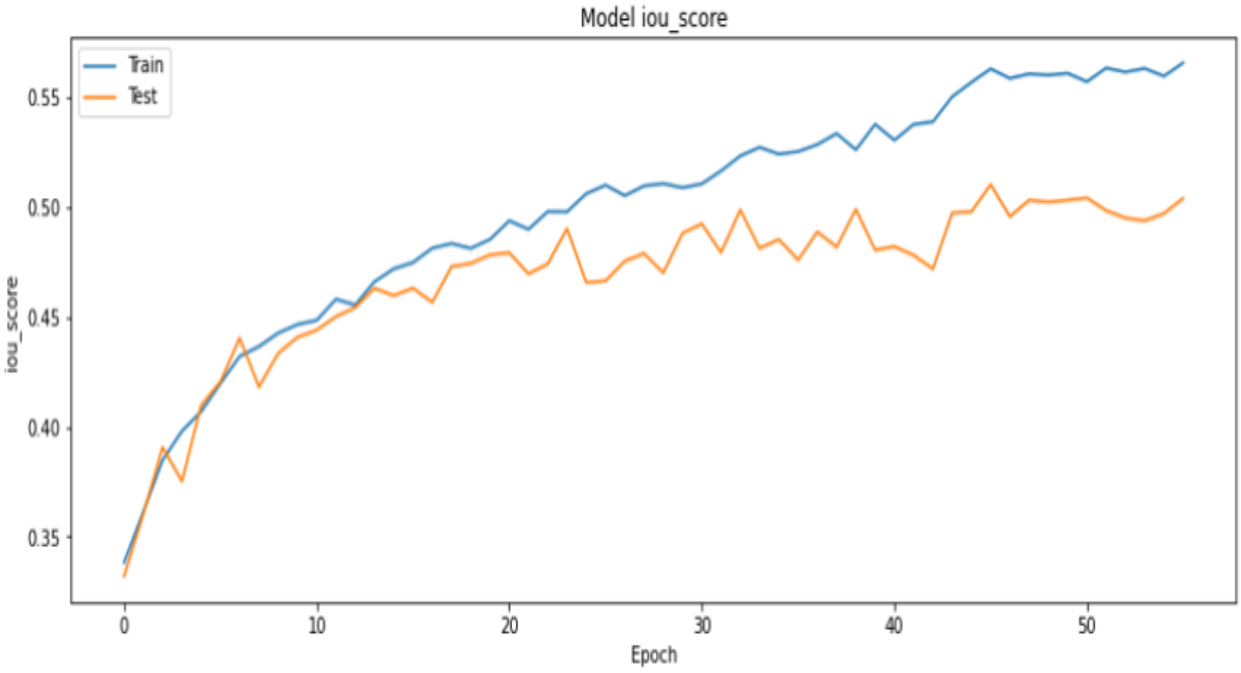


FIGURE 16 The mIoU of the FCN-16 Model with Dropout=0.5

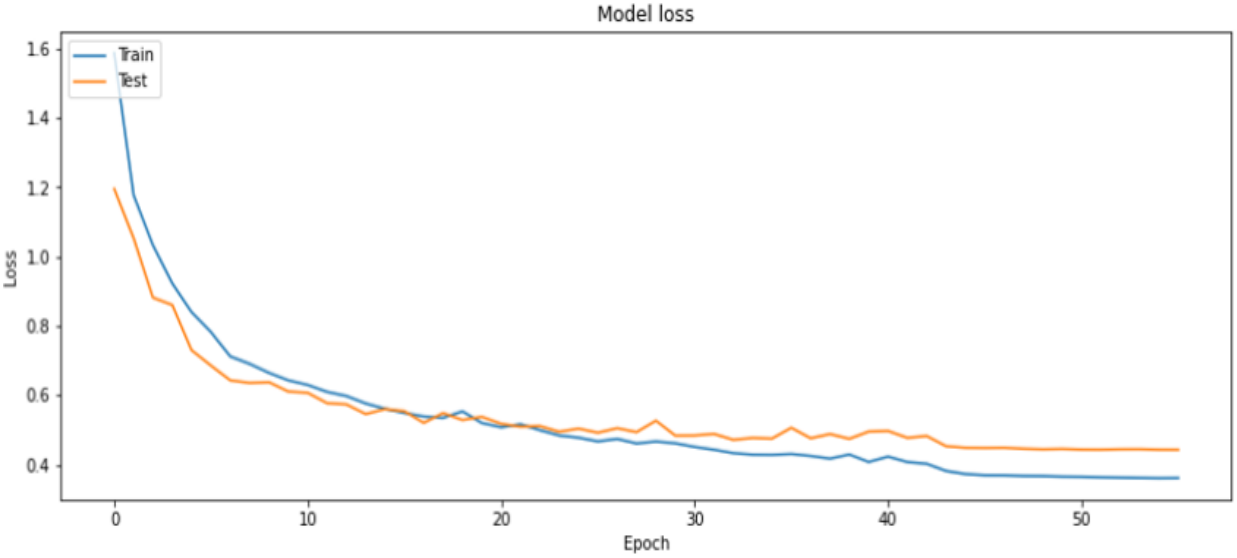


FIGURE 17 The Loss of the FCN-16 Model with Dropout=0.5

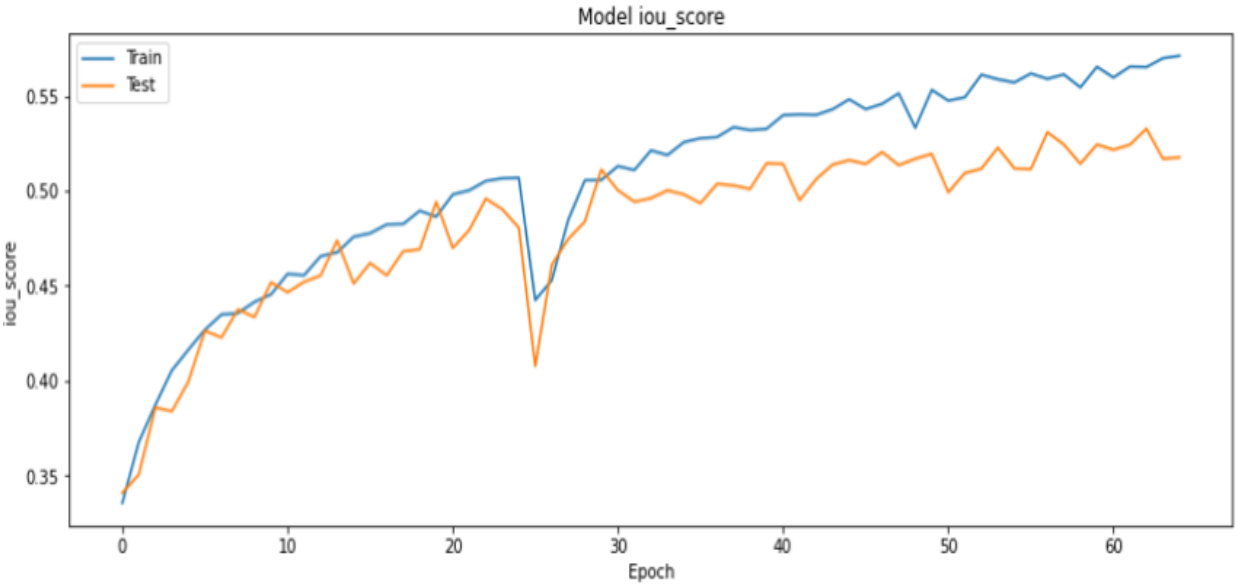


FIGURE 18 The mIoU of the FCN-8 Model with Dropout=0.5

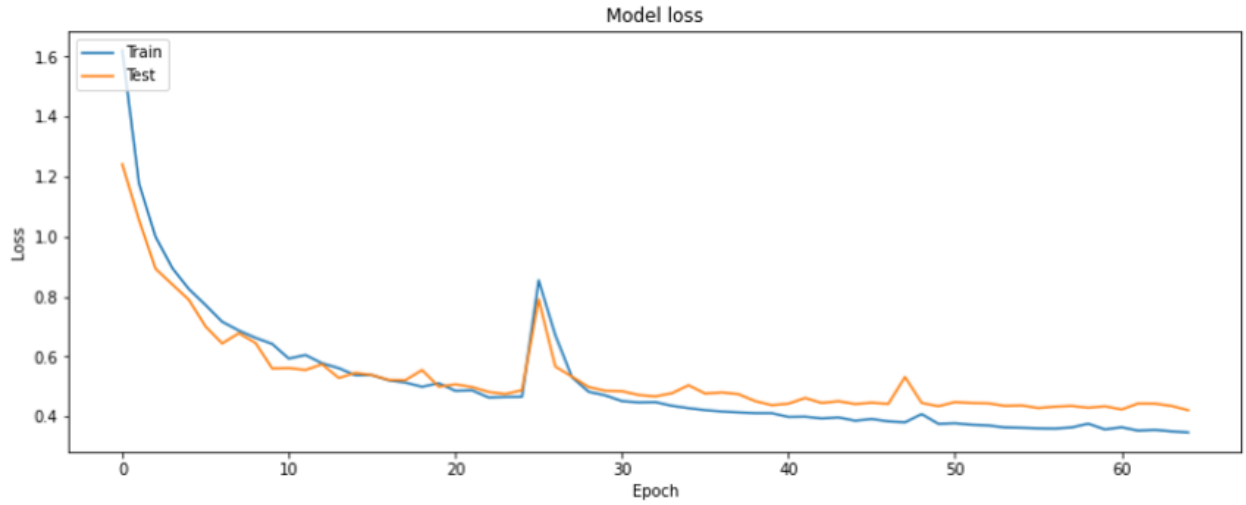
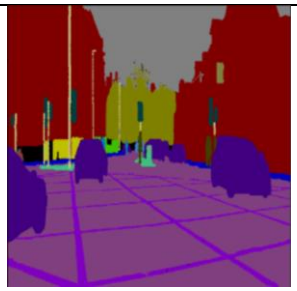
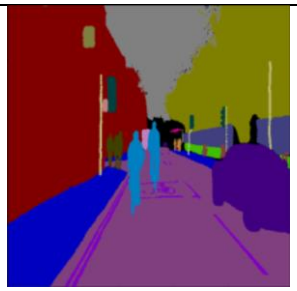
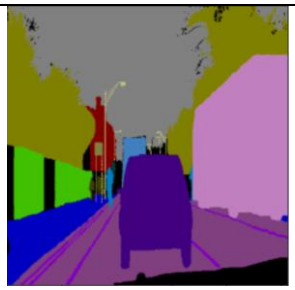
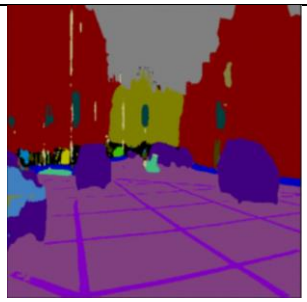
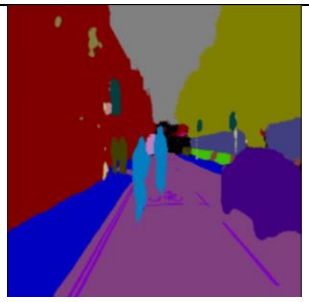
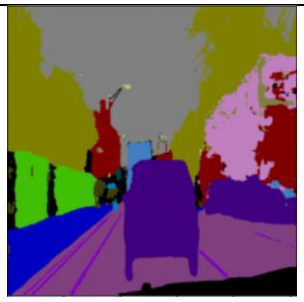
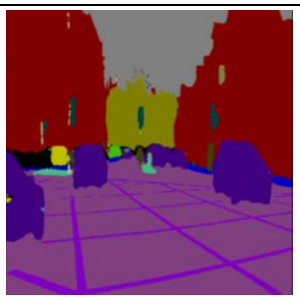
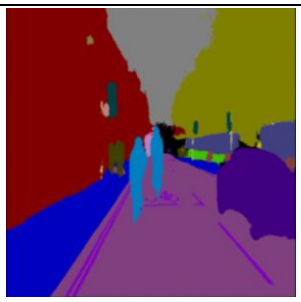
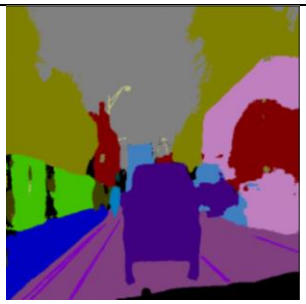


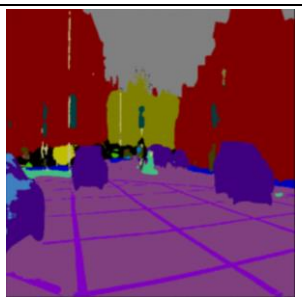
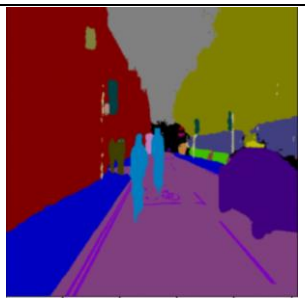
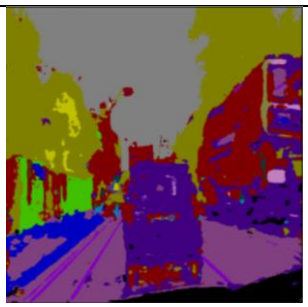
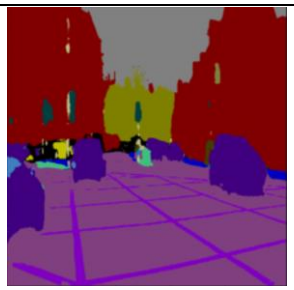
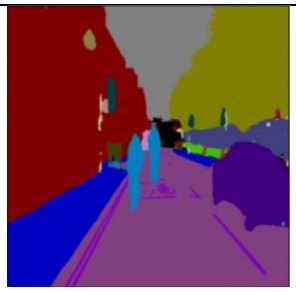
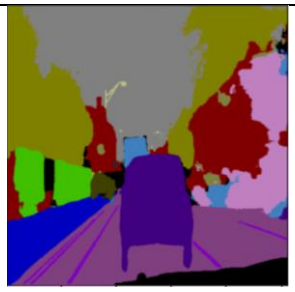
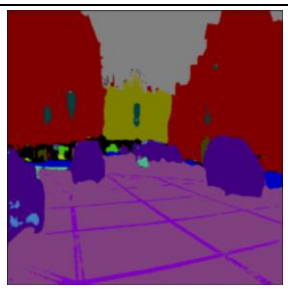
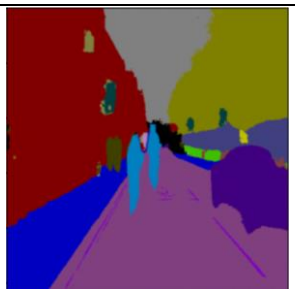
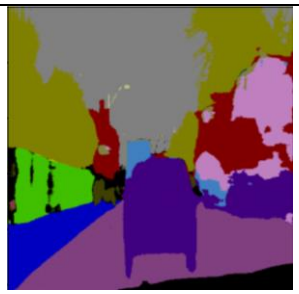
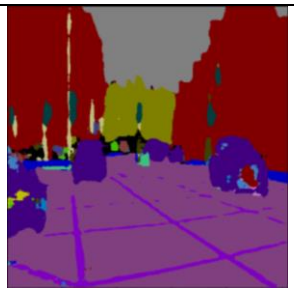
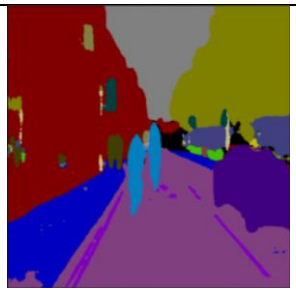
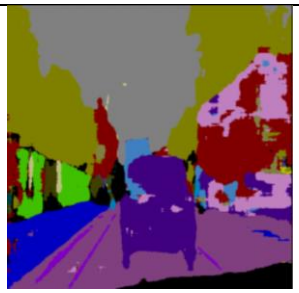
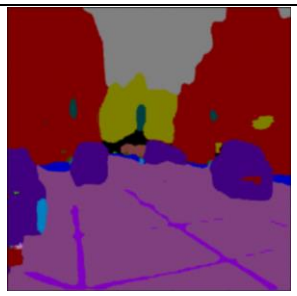
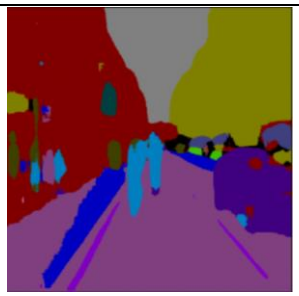
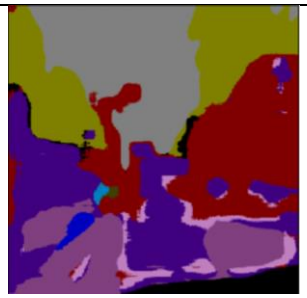
FIGURE 19 The Loss of the FCN-8 Model with Dropout=0.5

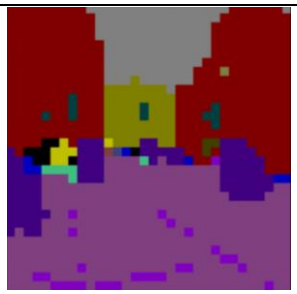
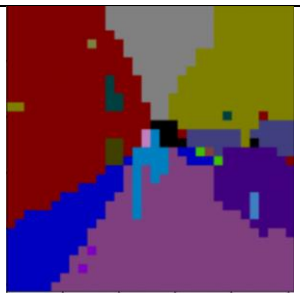
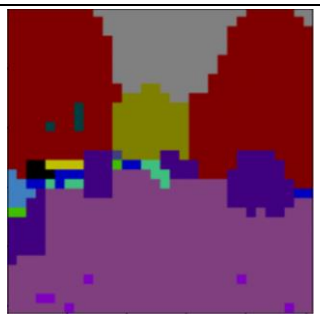
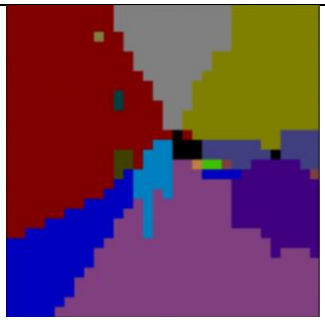
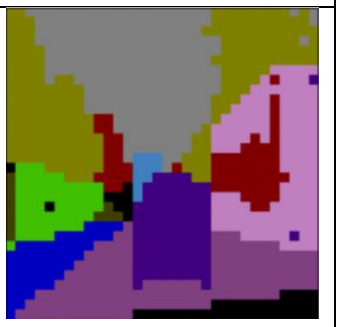
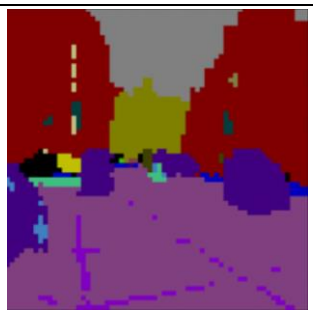
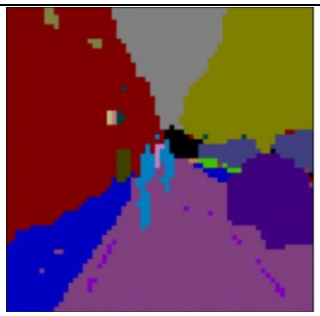
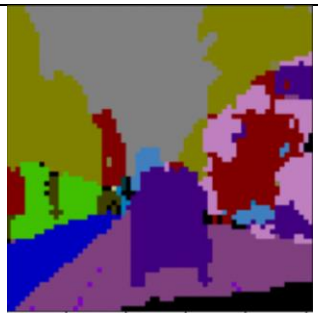
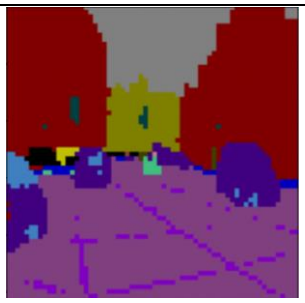
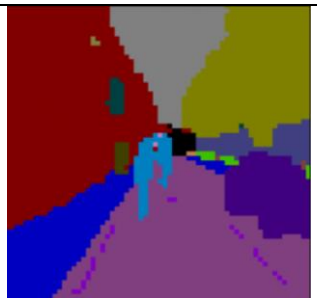
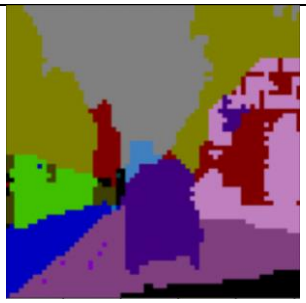
In Table 5, we have three samples of images with different scenarios and lighting conditions, with their corresponding ground truth labels and we tested each image on all eleven models. We can see that most U-Net models performed better in segmenting the images than FCN-16, FCN-8, and SegNet models and their variants. But the best segmentations were performed by the two Long U-Net models, as expected by the numerical analysis and comparison. They both perform very well in the first two images, due to the good

lighting conditions in both images, in the third image we can see that even though the lighting condition is poor but the Large U-Net models performed well in segmenting most of the objects, except for the bus, which can be traced to the lack of training images with busses in them, and the red segmentations instead of pink is due to the height of the bus, which the models are mistaking for a building.

TABLE 5 A Comparison Between the Resulted Segmentations

Original Image			
True Label			
Long U-Net Dropout=0.5			
Long U-Net Dropout=0.7			

U-Net ReLU			
Small U-Net ReLU			
Small U-Net Leaky ReLU			
SegNet			
SegNet Dropout=0.5			

FCN-16			
FCN-16 Dropout=0.5			
FCN-8			
FCN-8 Dropout=0.5			

The sensitivity analysis we performed, by building five U-Net models was for the mission of finding the best performing U-Net model. We first implemented the original U-Net model with ReLU, and built two other models with smaller feature channels one with ReLU and the other with Leaky ReLU as their activation functions, at this stage we could not see a difference in terms of mIoU, instead the accuracies for the models with smaller feature channels were lower and the losses were higher than the original U-Net. To see a bigger difference, the model architecture is needed to change, we build two deeper models

with two layers added on each path (Long U-Net), with different dropout rates, which learned the sophisticated features in the dataset better, the two models scored higher mIoU than the previous models, with the model with a dropout rate of 0.5 performing the best. To make a fair comparison with the three models: SegNet, FCN-16, and FCN-8, we implemented two variants of each of the three models, the first one is the original model and the second variant using the dropout regularization technique with dropout rate of 0.5, and the Long U-Net models still outperformed the three other models.

Chapter 5: Conclusions and Future Directions

As the adoption of autonomous vehicles with different levels of autonomy increases, the need for precise and accurate perception systems increases drastically to ensure the safety of the passengers, pedestrians, and the safety of the surrounding vehicles' drivers. Based on our extensive experiments presented in this project, we can conclude that U-Net can precisely classify and localize a wide range of objects in a complex driving environment and can outperform previously used well-known models in terms of mIoU, F1-Score, and accuracy.

In the future, we will train the U-Net models with different data augmentation techniques, with better computing power. To tackle the problem of poorly segmenting certain classes, such as the bus in the third image or the traffic poles, we will increase the weights on such classes and decrease the weights of other less necessary classes, to improve the overall performance of the model. We will also train and test different variations of the U-Net model on larger datasets and will compare it with other state-of-the-art semantic segmentation models. On the other hand, we will implement ensemble learning algorithms, by combining multiple state-of-the-art models together, to achieve the best performance possible. We will also take advantage of unlabeled datasets, to train unsupervised learning algorithms, those models can automatically learn complex road features with minimal human input.

References

[1] W. a. B. J. a. W. S. a. N. E. Zhou, "Automated Evaluation of Semantic Segmentation Robustness for Autonomous Driving," in IEEE Transactions on Intelligent Transportation Systems, 2018.

[2] J. S. L. a. T. H. Park, "Semantic segmentation with Improved Edge Detail for Autonomous Vehicles," in IEEE 16th International Conference on Automation Science and Engineering (CASE), Hong Kong, China, 2020.

[3] G. Cheng, J. Y. Zheng, and M. Kilicarslan, "Semantic Segmentation of Road Profiles for Efficient Sensing in Autonomous Driving," in IEEE Symposium on Intelligent Vehicle, 2019.

[4] M. Yoshioka, N. Suganuma, K. Yoneda, and M. Aldibaja, "Real-time object classification for autonomous vehicle using LIDAR," in International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), 2017.

[5] M. Feng, S. Hu, G. Lee and M. Ang, "Towards Precise Vehicle-Free Point Cloud Mapping: An On-vehicle System with Deep Vehicle Detection and Tracking," in IEEE International Conference on Systems, Man, and Cybernetics (SMC), Miyazaki, Japan, 2018.

[6] M. Gluhaković, M. Herceg, M. Popovic and J. Kovačević, "Vehicle Detection in the Autonomous Vehicle Environment for Potential Collision Warning," in Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 2020.

[7] J. Ciberlin, R. Grbic, N. Teslić and M. Pilipović, "Object detection and object tracking in front of the vehicle using front view camera," in Zooming Innovation in Consumer Technologies Conference (ZINC), Novi Sad, Serbia, 2019.

[8] L. a. Y. K. a. H. X. a. W. H. a. W. K. Sun, "Real-time Fusion Network for RGB-D Semantic Segmentation Incorporating Unexpected Obstacle Detection for Road-driving Images," IEEE Robotics and Automation Letters, vol. 5, no. 4, 2020.

[9] S. L. N. E. O. Y.G Naresh, "A Residual Encoder-Decoder Network for Semantic Segmentation in Autonomous Driving Scenarios," in European Signal Processing Conference (EUSIPCO), 2018.

[10] C. G. J. Y. Bike Chen, "Importance-Aware Semantic Segmentation for Autonomous Vehicles," IEEE Transactions on Intelligent Transportation Systems, vol. 20, pp. 137-148, 2019.

[11] K. L. Lim, T. Drage and T. Bräunl, "Implementation of semantic segmentation for road and lane detection on an autonomous ground vehicle with LIDAR," in International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI), 2017.

[12] S. Chen, Z. Zhang, R. Zhong, L. Zhang, H. Ma and L. Liu, "A Dense Feature Pyramid Network-Based Deep Learning Model for Road Marking Instance Segmentation Using MLS Point Clouds," IEEE Transactions on Geoscience and Remote Sensing, vol. 59, pp. 784-800, 2021.

[13] C. Lowphansirikul, K.-S. Kim, P. Vinayaraj and S. Tuarob, "3D Semantic Segmentation of Large-Scale Point-Clouds in Urban Areas Using Deep Learning," in 11th International Conference on Knowledge and Smart Technology (KST), Phuket, Thailand, 2019.

[14] X. Zhao, P. Sun, Z. Xu, H. Min and H. Yu, "Fusion of 3D LIDAR and Camera Data for Object Detection in Autonomous Vehicle Applications," IEEE Sensors Journal, vol. 20, pp. 4901-4913, 2020.

[15] K. Pranav and J. Manikandan, "Design and Evaluation of a Real-time Pedestrian Detection System for Autonomous Vehicles," in Zooming Innovation in Consumer Electronics International Conference (ZINC), 2020.

[16] L. Chen, Q. Ding, Q. Zou, Z. Chen and L. Li, "DenseLightNet: A Light-Weight Vehicle Detection Network for Autonomous Driving," IEEE Transactions on Industrial Electronics, vol. 67, pp. 10600-10609, 2020.

[17] H. Kim, Y. Lee, B. Yim, E. Park and H. Kim, "On-road object detection using deep neural network," in IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Seoul, Korea (South), 2016.

[18] J. Yang, C. Wang, H. Wang and Q. Li, "A RGB-D Based Real-Time Multiple Object Detection and Ranging System for Autonomous Driving," IEEE Sensors Journal, vol. 20, pp. 11959-11966, 2020.

[19] G. Prabhakar, B. Kailath, S. Natarajan and R. Kumar, "Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving," in IEEE Region 10 Symposium (TENSYMP), Cochin, India, 2017.

[20] P. Y. Hsu, M. L. Huang and H.-H. Chiang, "Trajectory of Prediction of Immediate Surroundings for Autonomous Vehicles Using Hierarchical Deep Learning Model," in IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), Yunlin, Taiwan, 2020.

[21] A. R. Fayjie, S. Hossain, D. Oualid and D.-J. Lee, "Driverless Car: Autonomous Driving Using Deep Reinforcement Learning in Urban Environment," in 15th International Conference on Ubiquitous Robots (UR), Honolulu, HI, USA, 2018.

[22] J.-G. Wang, L. Zhou, Y. Pan, S. Lee, Z. Song, B. S. Han and V. B. Saputra, "Appearance-based Brake-Lights recognition using deep learning and vehicle detection," in IEEE Intelligent Vehicles Symposium (IV), Gothenburg, Sweden, 2016.

[23] T. Okuyama, T. Gonsalves and J. Upadhay, "Autonomous Driving System based on Deep Q Learnig," in International Conference on Intelligent Autonomous Systems (ICoIAS), Singapore, 2018.

[24] Y. Zhang, P. Sun, Y. Yin, L. Lin and X. Wang, "Human-like Autonomous Vehicle Speed Control by Deep Reinforcement Learning with Double Q-Learning," in IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 2018.

[25] G. ÖZTÜRK, R. KÖKER, O. ELDOğAN and D. KARAYEL, "Recognition of Vehicles, Pedestrians and Traffic Signs Using Convolutional Neural Networks," in 4th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Istanbul, Turkey, 2020.

[26] H.-T. Tseng, C.-C. Hsieh, W.-T. Lin and J.-T. Lin, "Deep Reinforcement Learning for Collision Avoidance of Autonomous Vehicle," in IEEE International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan), Taoyuan, Taiwan, 2020.

[27] Y. Fu, C. Li, F. R. Yu, T. H. Luan and Y. Zhang, "A Decision-Making Strategy for Vehicle Autonomous Braking in Emergency via Deep Reinforcement Learning," IEEE Transactions on Vehicular Technology, vol. 69, pp. 5876-5888, 2020.

[28] S. Shi, X. Wang and H. Li, "PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 2019.

[29] G.-H. Lin, C.-H. Chang, M.-C. Chung and Y.-C. Fan, "Self-driving Deep Learning System based on Depth Image Based Rendering and LiDAR Point Cloud," in IEEE International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan), Taoyuan, Taiwan, 2020.

[30] G. Pang and U. Neumann, "3D point cloud object detection with multi-view convolutional neural network," in International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 2016.

[31] V. Badrinarayanan, A. Kendall and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481-2495, 1 Dec. 2017.

[32] J. Long, E. Shelhamer and T. Darrell, "Fully convolutional networks for semantic segmentation," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431-3440.

[33] Segmentation and Recognition Using Structure from Motion Point Clouds, ECCV 2008 Brostow, Shotton, Fauqueur, Cipolla.