Lane Detection Model using Deep Learning Algorithms

Juan Antolín Tello del Rosal
Universidad Pontificia Comillas ICAI
August 25, 2020

Resumen— Una de las partes más importantes y necesarias para conseguir un satisfactorio modelo de Conducción Autónoma y de sistemas de ayuda a la conducción es la detección de la carretera y sus límites. El objetivo de este proyecto consiste en la creación de un Modelo basado en Deep Neural Networks, el cual recibirá como entrada los fotogramas captados con una cámara localizada en el vehículo, y como salida la representación de la carretera y sus límites. Muchos modelos han sido propuestos para la detección de las líneas de la carretera y sus límites, pero la gran mayoría de ellos se basan en la detección de carretera en entornos urbanos y en autopistas, donde la calzada esta hecha de asfalto. El modelo propuesto desea poder detectar la carretera en ambientes más difíciles, capaz de detectar la carretera en calzadas no hechas de asfalto, como calzadas de adoquines. El modelo propuesto se basa en la segmentación semántica binaria de cada pixel de la imagen para distinguir cada pixel entre carretera y no carretera. La arquitectura elegida para el modelo es una Full Convolutional Neural Network, que no contiene ninguna capa conectada y basa su predicción en una sucesión de Convoluciones y Deconvoluciones para generar el mapa de píxeles final que será el resultado de la detección de la carretera. Los resultados obtenidos son consistentes, y el modelo es capaz de detectar la calzada del circuito en cualquier situación de una manera robusta y continua.

Abstract— One of the most important and necessary parts to achieve a successful Autonomous Driving model and driving assistance systems is the detection of the road and its limits. The objective of this project is to create a Model based on Deep Neural Networks, which will receive as input the frames captured with a camera located in the vehicle, and as output the representation of the road and its limits. Many models have been proposed for the detection of road lines and their limits, but the vast majority of them are based on the detection of roads in urban environments and on highways, where the road is made of asphalt. The proposed model wants to be able to detect the road in more difficult environments, capable of detecting the road on non-asphalt driveways, such as cobblestone driveways. The proposed model is based on the binary semantic segmentation of each pixel of the image to distinguish each pixel between road and non-road. The architecture chosen for the model is a Full Convolutional Neural Network, which does not contain any connected layer and bases its prediction on a succession of Convolutions and Deconvolutions to generate the final pixel map that will be the result of road detection. The results obtained are consistent, and the model is capable of detecting the road surface of the circuit in any situation in a robust and continuous way.

Key Words: Deep Learning , Lane Detection, Convolutinal Layer, Semantic Segmentation, Encoder-Decoder.

1. Introduction

Autonomous driving has become a reality during the last years. Nowadays it is easy to see the application of autonomous driving in many different environments. From from automatic robots in a warehouse, through delivery drones, to fully autonomous vehicles capable of moving freely on public roads without the need for any action by the driver.

The main purpose of the autonomous driving is to create vehicles capable to fulfil different task without human intervention. As final objective, autonomous driving wants to make peoples life easier and more safe.

Lane detection is one of the most important parts of the autonomous vehicle ecosystem. It determines the road that the vehicle must follow and a good lane detecting will provide a huge impact on the autonomous vehicle overall performance. During the last years, with the development of Machine Learning and the improvement of Neural Networks and its architectures, there has been a drastic change on how image processing is being approached. The old standard methods based on computer vision are now being replaced by these new Neural Network based models, which are more powerful and achieve a better results, but having a higher computational power need.

The main motivation of this project is to avoid human intervention during this testings and avoid putting human lives at risk sometimes, due to the possibility of accidents during this testings and to avoid material loses due to the vehicle accidents.

The Road Lane Detection final be implemented in a vehicle, specifically a FUSO Canter truck, located in the Kitsuregawa Testing Truck facility. The trucks are used in enduring and durability testing. The circuit consist of two main straight roads with some slight curves along the roads and two U-turns or sharp turns at the end of the road, creating a simple circuit to test the vehicles.

The biggest challenge for the development of this model is the type of road present in the circuit. The Testing facility circuit counts with two main types of road:

- Smooth Road: The first type of road, which will be called "Smooth road" from now on, consist in a normal, asphalt road, where can be find in highways and secondary roads all around Japan.
- Rough Road: The second type of road, which will be called "Rough road", is not made of asphalt. Instead, it is made of paving stone, making the road rougher and not as uniform as the Smooth road. The Rough road also will create some difficulties to the video streaming. Since the road is not made of asphalt, the camera located in the truck will receive all the vibrations the vehicle will experience, making the video footage not steady during this type of road.

Is in this Rough Road scenarios where the models will not perform as good as in other scenarios due to the difference between the training data road type and the Rough Road type.

2. State of the Art

Several types of detection and prediction models have been proposed over the last 20 years. These models can be classified in two main categories, the traditional methods and the deep learning methods.

Traditional Methods: These so called traditional Lane Detection methods based the detection of the road by primitive elements such as gradient, color and texture of the image. Two main categories can be distinguish, the Geometrical modeling methods and the Energy minimization methods.

The Geometric modeling methods based their Lane Detection in two steps, first edge detection and then line fitting. Edge detection was achieved using different gradient filters to detect big changes in the color and texture of the input image. Canny Edge Detector [1] model and Gabor Filters [2] are commonly used to generate the edge detection. For line fitting the most used algorithm is Hough Transformation [3]. This algorithm takes the edges generated and transforms them in polynomial functions to be used as the representation of the road. These models kept being the base of the first autonomous vehicles right before starting to implement the new Deep Neural Network models.

Deep Learning Methods It was with the introduction and the use of Deep Learning algorithms when the Lane Detection topic made a big leap into the real application of Autonomous Vehicles.

A huge variety of deep learning Lane Detection models have been proposed in the past several years. These models can be categorized in three main methods.

Encoder-Decoder Convolutional Neural Network: The encoder-decoder CNN architecture is the main type of network to develop Semantic Segmentation algorithms. These end-to-end architectures are built to detect and separate all elements from the input image in different categories, depending on the desired output of the model.

The main structure of these encoderdecoder architectures are the 2D Convolutional layers. These layers, unlike the regular full connected hidden layers of the regular Neural Networks, do not connect all elements of the image and processes them. These layers take a threedimensional input, typically an image with three color channels. Then the image is scanned passing a convolution kernel over the image. This kernels works as a filter, inspecting a small window of pixels at a time, for example 3 x 3 in size, and moving the window until they have scanned the entire image. The convolution operation calculates the dot product of the pixel values in the current filter window with the weights defined in the filter.

Using these layers as base, the encoder-decoders take the image as input and the different convolutions gather all the information from all different aspects and features of the input image and use them to create a pixel segmentation map, which will be the classification map of the image.

The main applications of Semantic Segmentation are in autonomous vehicles, human-computer interaction, robotics, and photo editing/creativity tools. There has been created Semantic Segmentation models to focus on specific tasks, like the UNet model [4].

This model was developed to create segmentation of neuronal structures in electron microscopic stacks. Several architectures have been used to create semantic segmentation models to be used in Vehicle road detection. The main references used to develop the project were the SegNet architecture and the ERFNet architecture.

The SegNet architecture proposed in 2015 [5], which is a Semantic Segmentation model used to detect the different elements which conform the road scene . This Neural Network architecture is based on a Full Convolutional structure, being an end-to-end full convolutional network, where the Decoder structure is a mirrored version of the Encoder.

The ERFNet model is another Semantic Segmentation approach to be used in vehicle detection, proposed in 2017 [6].

Thestructure of the ERFNet model consisted of a Encoder-Decoder architecture using a specific type of Convolution layers instead of the typical 2D Convolution. These layers are the so called Factorized Resnet Modules with Dilation, which they allow or very deep models to be created without as much risk of vanishing/exploding gradients. Each module include one-dimensional dilated convolutions as base and adding dilated convolutions to give the layers in the network a lot of context. This allowed to have a reduced output time while preserving the input image resolution.

Recurrent Neural Networks + CNN :The CNN+RNN models include apart from the Convolutional standard architectures, some recurrent Neural Networks elements to improve the accuracy of the prediction by including the previous predictions as part of the prediction process.

The main reference of CNN+RNN network for future steps that the project is the model proposed Qin Zou et al, in 2018 with their Robust Lane Detection from Continuous Diiving Scenes. [7]

This model architecture consist of three differentiated parts. Apart from the encoder and the decoder, the architecture includes a ConvLSTM block to treat the outputs feature maps from the encoder. In our network, the input and output size of the ConvLSTM are equal to the size of the feature map produced by the encoder. The size of the convolutional kernel is 3 by 3. The ConvLSTM is equipped with 2 hidden layers, and each hidden layer has a dimension of 512. The main objective of the ConvLSTM is to forget the unimportant information for the feature maps extracted from the CNN and remember the essential features from the previous predictions.

3. Methodology

The development of the project has been divided into two main different phases, the Dataset creation and the Model Creation.

Datasets:For the development of the different models several datasets have been used. There are two main types of datasets used in the development of the models, depending if the model is a Whole Road model or an EgoLanes model. For the EgoLanes models there is only one dataset used, the TuSimple dataset. The TuSimple Dataset is a dataset owned by the company TuSimple, a self-driving truck company focused on the development of selfdriving heavy-duty trucks. This dataset was released as part of the TuSimple Lane Detection Challenge. It consist of 3626 video clips of 1 sec duration each, with each video clip containing 20 frames. The quality of each frame is 1280 x 720 pixels, in RGB format. From these frames the last frame is labeled, each line labeled on its own. The dataset can be obtained for free as a public download form the Github page of TuSimple [8].

To increase the size of the training data, each image and label of the dataset has been preprocessed by image transformation methods. Each image and label was mirrored and tilted 4 degrees in each direction to increase the amount of images of the dataset from 3626 up to 14504 images and labels. Since the quality of the images were really high, all images were resized to 160 by 80 pixels. The dataset was compressed into two different pickle files, one for the input images and one for the ground truth labels.

The following image show an image representation of the TuSimple Dataset:

For the Whole Road models, two different dataset were used for training. The first model was the Baseline dataset, which was created by Michael Virgo [9] [10] and it consist of a total of 2127 images and labels from different . To increase the number of data same image transformation methods as used in the TuSimple dataset were applied to this footage, increasing the total amount of images up to 12762 images and ground truth labels.

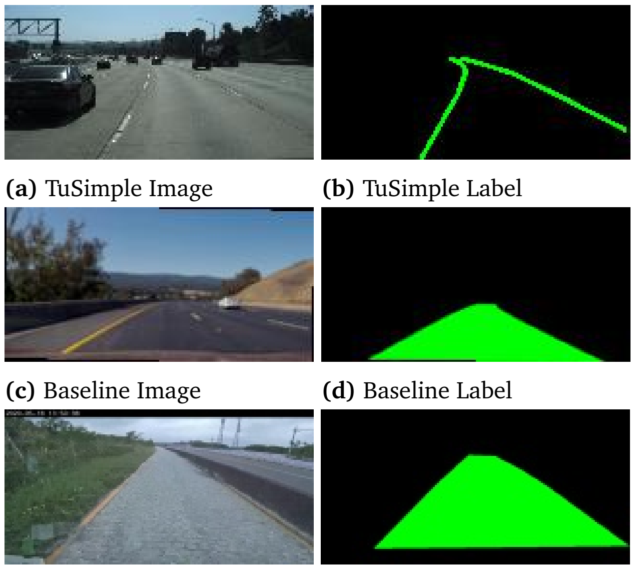
The FUSO dataset was created with the purpose to increase the performance of the models in Rough Road scenarios. This dataset was created using a labeling tool designed in Python, where it took from a video of the FUSO testing circuit frame by frame and manually create the ground truth representation. A total of 504 images and labels in different scenarios where created. This dataset was designed to be used by the Whole Road models.

Apart from the Training datasets, two more datasets were created. These datasets were used to test the different models performance. The test datasets are two, the Test dataset, which is used to test the Whole Road models and the TuSimple Test dataset, created to test the EgoLanes models. These datasets contains a total of 240 images and labels of all different scenarios of the FUSO testing circuit. Same as the FUSO dataset, these dataset were created using the Labeling tool taking frames from prerecorded videos from the Kitsuregawa circuit.

Models: Several models were created during the development of the project. All the models were based in two main Neural Networks architectures.

SegNet Architecture: the SegNet architecture, as explained in the State of the Art section, the SegNet architecture is a encoder decoder architecture used for semantic segmentation problems. The model consist in a sequence of convolution layers with RELU activation, followed by a Pooling layer to reduce the size of the amount of parameters needed. It uses Max Pooling to select the maximum values of the previous layer in each kernel. This process is then copied again, using convolutional layers and polling layers.

After this Convolution + Pooling process is finished, a mirrored version of Upsampling and Deconvolution is created to recreate the output pixel map segmentation to the same size as the input image. This dropout helps to prevent overfitting the layers weights during the training period.



(a) TuSimple Image **(b)** TuSimple Label

(c) Baseline Image **(d)** Baseline Label

(e) FUSO Image **(f)** FUSO Label

Figure 1: Examples of the different datasets. In descendt order, TuSimple Dataset, baseline Dataset, FUSO dataset

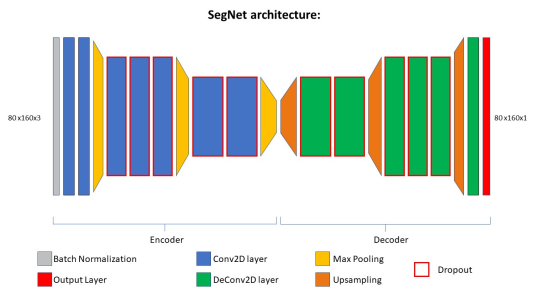


Figure 2: SegNet architecture

ERFNet Architecture: This model consist of a combination of 2DConvolutional layers and max pooling layers which form the Encoder structure, and a series of 2D Deconvolutional layers and upsampling layers forming the Decoder structure. All Convolution and Deconvolution process have the ReLU function as activation function, and all layers use padding to maintain the same input size along the whole Neural Network.

The first layer of the Neural network, same as used in the SegNet architecture, is a batch normalization layer. Right after the normalization a max pooling layer downsizes the input image and passes it to the firs Convolutional layer of input size (40, 80, 3). After this first convolution another max pooling is done, followed by five 2D Convolutional layers with 32 kernels each, and an input size of (20, 40) and all of the layers after the second max pooling will have dropout to avoid overfitting of the model during the training stage of it. The last part of the decoder, the inner layers, consist of eight 2D Convolutional layers with 64 kernels each. The input size of these inner layers is (10,20) and their focus is to get all the deep insights from the image. All these inner layers will have dropout of 0.2 to avoid overfitting the model.

The decoder structure consists of differentiate three parts divided by the upsampling layers. Right after the inner layers part of the decoder a upsampling layer with a pool size of (2,2) leads to three Deconvolutional layers, another upsampling followed by another three deconvolutional layers. These Deconvolutional layers are have the exact input size as the Convolution layers. The last two layers of the Decoder are a Deconvolutional layer with 16 kernels right before the Final layer, which has the output size of (80, 160,1).

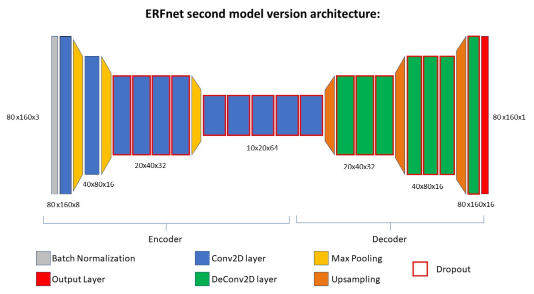


Figure 3: ERFNet reduced architecture

Whole Road Models:

SegNet Base model: The first model created uses the SegNet architecture as base, and it was trained with the Baseline dataset. It is a WHole Road model which has not been trained using FUSO footage and will be used as the Baseline model.

Retrained Model: SegNet Base model did not perform correctly in Rough Road scenarios. The decision taken was to use Tune Fitting using the FUSO dataset, updating the different model weights to make them able to understand the Rough road.

ERFNet Reduced Model: Whole Road model trained with Baseline + FUSO datasets and uses the ERFNet Reduced architecture.

EgoLanes Models:

EgoLane model trained with the TuSimple dataset and SegNet architecture. Will be used as baseline for the EgoLanes models.

ERFNet TuSimple: EgoLane model trained with the TuSimple dataset and ERFnet Reduced architecture.

4. Results and Model Comparison

Two main methodologies have been used to measure the performance of the models ant compare the results obtained.

The first part will be a visual comparison of the output of all models compared to the baseline model. In the coarse level, the model is expected to predict the Whole road or the two Egolanes, depending on the model, correctly. Two detection errors should be avoided in the processing of lane detection. The first one is missing detection, which predicts the true lane objects in the image as the background, and the second is excessive detection, which wrongly predicts other objects in the background as the lanes. Both these two detection errors will cause the inconsistency of the width of the lane and the number of lanes detected by the EgoLanes models.

Qualitative Visual method:

The second validation technique will consist in a quantitative calculation based on several metrics, tested with the Test dataset.

The different visual outputs of the model can be seen in the Figure 4.

Whole Road Models: The Baseline model doesnt not perform well for all scenarios. It s able to give a decent and dense pixel map in Smooth Road and Clear Day but is not able to detect the road during the Rough Road Scenarios. After Tune Fit it with the FUSO dataset, the Retrained model is capable to detect the road during the Rough Road scenario, generating a decent pixel map, but not dense enough to detect the whole road. It is possible to observe that the retrained models does not perform properly in Night Scenarios.

On the other hand the ERFNet Reduced model is able to generate a dense pixel map covering the whole road and not giving excessive predictions in almost all scenarios.

EgoLanes Models: The Egolanes models do not perform really well in general.

The SegNet TuSimple model has a very low-density prediction lines, making difficult to detect the limits of the road. Although the model is able to give an accurate prediction in Smooth Road scenarios when the marked lines are present, during Rough Road and Night scenarios the model gnerates erratic predictins which do not follow the margins of the road and generates several in the middle of the road besides the egolanes.

The ERFNet TuSimple model outperforms the SegNet TuSimple model in all scenarios, generating a more dense pixel map prediction which cover the whole lines lenght. However, same as the SegNet TuSimple model, the output performance in Rough Road and night scenarios is very eratic and not reliable and robust.

Quantitative analysis method: This quantitative analysis will serve as a base to determine effectively which model has the best performance during all different scenarios of the circuit and will give a better way than just a visual recognition of the different outputs to determine the best model performance.

To Test the model, the already commented Test Dataset was used. For the TuSImple models the TUSimple Test dataset was used. The main metric to measure the accuracy of the models is the Intersection over Union or Jaccard metric [11]. This method is commonly used to measure the accuracy of Semantic Segmentation models. The intersection $(A \cap B)$ is comprised of the pixels found in both the prediction mask and the ground truth mask, therefore, the true positives.The union $(A \cup B)$ is simply comprised of all pixels found in either the prediction or target mask.

$$\mathbf{IoU} = \frac{(\mathbf{A} \cap \mathbf{B})}{(\mathbf{A} \cup \mathbf{B})} \tag{1}$$

As seen in Table 1, it is possible to appreciate the big difference between the Whole Road models and the EgoLanes models. This main difference can be caused by the different dataset used for each type of model. Also, the Egolanes models had a lower number of pixels in each ground truth label to compare, plus both Egolanes model had several excessive prediction of lanes which were not part of the Egolanes.

Even the IoU metric is commonly used in Semantic Segmentation problems, a slightly error on the classification of the pixels can lead to drop the accuracy levels fairly easy. To have a better understanding on how each model performs, new metrics are used. Precision and recall are employed as two metrics for a more fair and reasonable comparison, which are defined as:

$$\mathbf{Precision} = \frac{\mathbf{True\ Positive}}{\mathbf{True\ Positive + False\ Positive}} \tag{2}$$

and

$$\mathbf{Recall} = \frac{\mathbf{True\ Positive}}{\mathbf{True\ Positive + False\ Negative}} \tag{3}$$

Since the Road Lane Detection problem is a binary classification problem, the lane is classified as positive class while the background is classified as negative class. True positive values refer to all pixels classified by the model as part of the road and are labeled as road in the ground truth. The False positives are the pixels categorized as part of the road but are are labeled as background in the ground truth. Finally, the False Negatives pixels are pixels categorized as background while they are labeled as road in the ground truth.

In summary, Precision gives the proportion between the pixels correctly categorized as road divided by the total amount of pixels categorized as road, and Recall gives the proportion between the pixels correctly categorized as road and the true road pixels. For example, if the models generates an output with a really low dense pixel map, but all the pixels are categorized as road, the Precision of the model will be almost 100%, but the Recall will have very low values.

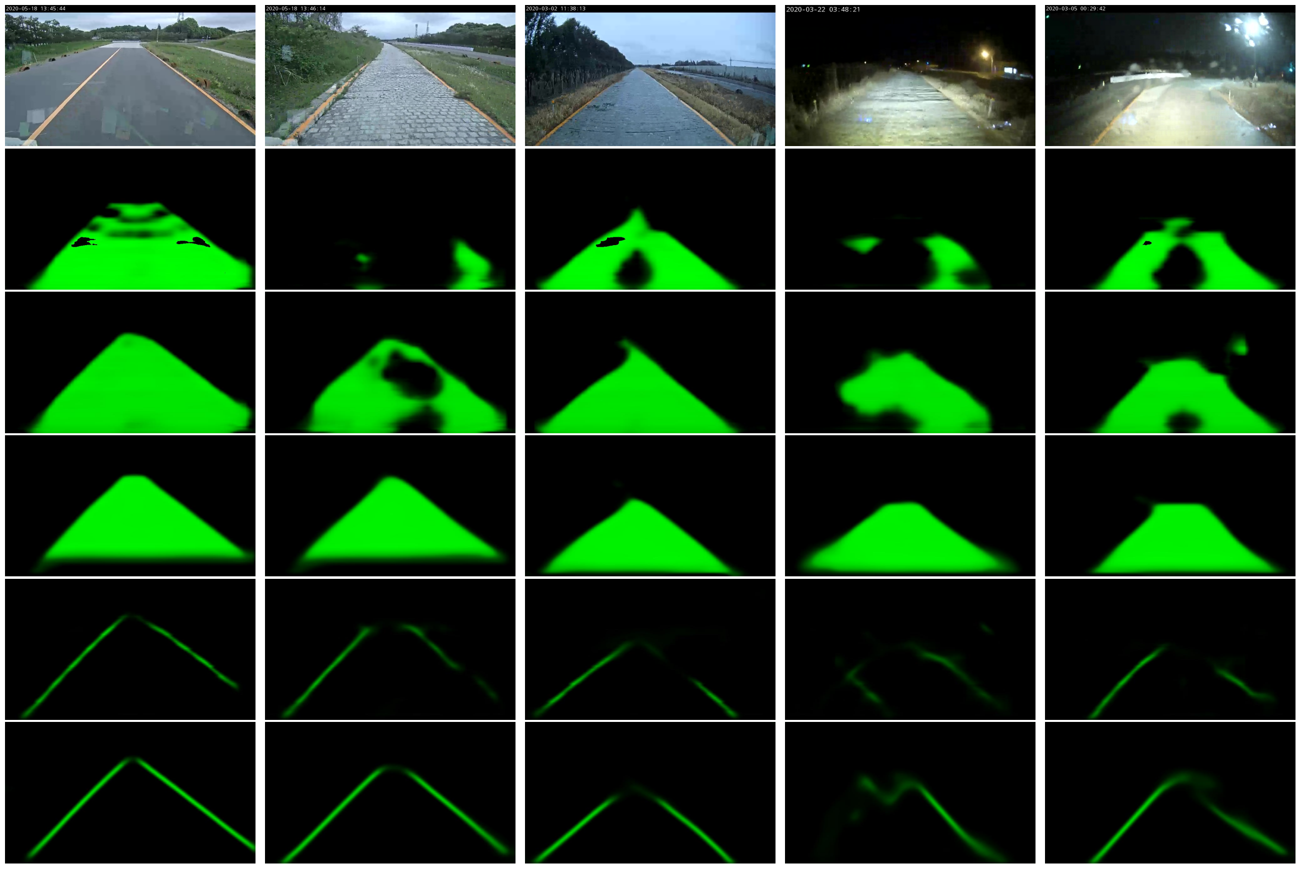


Figure 4: Visual comparison of the Lane detection models in different scenarios. Row 1: Original input image, Row 2: SegNet Baseline output, Row 3: Retrained Baseline output, Row 4: ERFNet Reduced output, Row 5: SegNet TuSimple output, Row 6: ERFNet TuSimple output

Since the Precision and the Recall only represents one aspect of the performance of the model, it is necessary to find a way to measure the effect of both metrics at the same time. To solve this problem, the last metric proposed to compare the models performance is the F1 or Dice metric.

The F1 score is the harmonic mean of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall). The F1 score is also known as the Sørensen–Dice coefficient or Dice similarity coefficient (DSC)." [12]. This coefficient will take into account both Precision and Recall to calculate the accuracy of the output prediction. The formula to calculate the F_1 is the following:

$$\mathbf{F\_1} = \mathbf{2} * \frac{\textbf{(Precision * Recall)}}{\textbf{(Precision + Recall)}} \tag{4}$$

In the Table 1 can be found the values for the metrics obtained from all the models.

The results obtained shows a big difference between the Whole Road models and the EgoLane models. This main difference can be caused by the different dataset used for each type of model. Also, the Egolanes models had a lower number of pixels in each ground truth label to compare, plus both Egolanes model had several excessive prediction of lanes which were not part of the Egolanes. These two reasons made the values of the metrics of the Egolanes model to drop below 0.5 in all metrics. Even though the low results, it is visible the big increase in the Recall of the ERFNet TuSimple Based model vs the SegNet TuSimple model.

On regard of the Whole Road models, the Baseline model starts with a IoU of 0.7 and an F1 of 0.81. Being the first model developed the results are good. However, these high values of IoU and F1 contrast with what have been seen in the visual comparison and the visual representation of the output. The main reason of these values can be because the size of the label ground truth compared to the total size of the image. This will increment the values of the metrics even though the models do not perform in a very high accuracy.

Despite this bias on the metrics, it is possible to appreciate the increase of the values of all metrics in both the Retrained model and the ERFNet model compared to the Baseline model.

The Retrained model has a slightly lower Precision compared to the Baseline model, which means that it has a little more false positives pixels than the Baseline model, but there is a huge increase in the Recall performance, with 0.13 more than the Baseline models recall. this mean that the Retrained model output has less False Negatives, and the predicted pixel map covers a bigger area of the road despite not being as precise as the Baseline Model. Taking the IoU and the F1 metrics is possible to see that the Retrained model is 7% more accurate than the Baseline model.

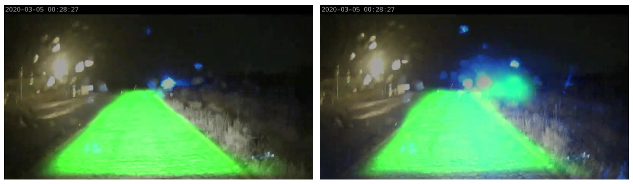
From all the models Tested during the development of the project, the ERFNet Reduced model has obtained the best results in all the metrics available. It has a 0.9 Precision and a 0.96 Recall, 0.07 and 0.16 points higher than the Baseline model respectively, and it has a F1 of 0.9283, being the first model to surpas the 0.9 value in this metric. The ERFNet model also has the best overall accuracy of all models, with a 0.8703 IoU, 10% higher than the Retrain model and a 17% higher than the Baseline model.

Final Results: After the Qualitative analysis and the Quantitative metrics comparison of the different models, it can be observed that the ERFNet model, both using the Quantitative and the Qualitative Visual methods, is the best performing model in all different Scenarios proposed in the Kitsuregawa Testing Circuit.

Using the visual Qualitative methodology to determine the best model, the ERFNet reduced model has the best performance of all the models tried. The model is able to predict the road in a accurate way and has both good detection and low excessive prediction of the road, except in some night scenarios. It is noticeable to add that the Final model also has a good performance in continuous frames, and the prediction between consecutive frames are very similar.

However, this continuous prediction is not always present. In the Qualitative testing videos it was possible to observe during the Night Scenarios that, due to changes of lightning (for security reasons there is in the Kitsuregawa Testing Circuit a intermittent blue light which interferes with the lights of the truck and changes the lightning of the road), the prediction between some frames changes drastically and leads to creation of a very different pixel map of the road.

These effect can be seen in the following images:



(a) First Frame Output **(b)** Second Frame Output

Figure 5: Consecutive frames obtained from the Night Raining Video Testing of the ERFNet model.

Despite the some minor problems, such as the sporadic excessive prediction seen in Figure 5, the model performs in a robust way, being able to continuously detect the Road in any possible scenario, even the most adverse ones.

5. Conclusions

The election of the SegNet architecture as the baseline of the model was because previous models based on SegNet had performed in good conditions. The main reason the Baseline model did not performed correctly during the Test phase was mainly due to the difference between the Test road and the training images road. This difference was more clear in the Rough Road scenarios and and in night situations. One possible option that was presented during the development of the project was to expand the Neural Network, adding some layers at the beginning and at the end of the Network and train them with the Circuit images. The main issue observed was that the it was not possible to transfer the weights of the different Convolutional layers from the SegNet Baseline model, making that approach not viable and at the end opted for the Transfer Learning and retrain the Baseline model with some inside FUSO footage.

Since the tune fitting helped to improve the output performance, it was clear that introducing as training images footage from the Kitsuregawa testing facility circuit helped to improve the model, it was decided to train the next iteration of models with both the Michael Virgo dataset and the FUSO dataset.

Even though the improvement on the performance of the model after the introduction of retraining the model with FUSO data, it was noticeable that the model was still not able to gather all the different factions of the road and it was visible that some information were being lost in the network. One possible reason of this was the inner layers of the model. The center of the SegNet architecture was a final Max Pooling followed for a Upsampling layer. It was possible that in that dimensional reduction of the outputs of the inner 2D Convolutional layers some valuable information of the road was missing.

This was the reason to create the new architecture based on the ERFNet layer structure, the ERFNet Reduced architecture. The main difference was the new architecture did not have a mirrored Encoder-Decoder structure, and avoided the last centered Max Pooling Upsampling layers. Also, adding more inner layers inside the architecture helped to generate a more robust output throughout all different scenarios.

Regarding on how to test this kind of semantic segmentation models, the final results of the metrics are strongly defined by the real size of the pixel map that wants to be predicted. The enormous gap between

Models Metrics Evaluation Results

Model	Precision	Recall	IoU	F1
Baseline Model	0.8359	0.8054	0.7021	0.8146
Retrained_5 Model	0.8227	0.9357	0.7798	0.8709
ERFNet_reduced Model	**0.9026**	**0.9603**	**0.8703**	**0.9283**
TuSimple Base Model	0.4705	0.2361	0.1875	0.3071
ERFNet_TuSimple Model	0.5137	0.5030	0.3528	0.5015

Table 1: Model Test metrics

the values obtained for the EgoLanes models and the Whole Road models were too big that the models could not be compared between them. The area of the ground truth labels of the Whole Road models covers almost a third of the image, making easier to have a better percentage of predicted pixels inside the testing label. On the other hand, the EgoLanes models ground truth testing labels only covers a really small amount of the image, and making the values of the metrics to plummet compared to the Whole Road models. Therefore, the testing method needs to be improved to be able to compare the accuracy of both types of models.

In conclusion, the Final model deployed at the Autonomous Testing Truck is capable to detect the Road of the Testing Circuit in all different scenarios proposed at the beginning of the project, facing some small issues at Night scenarios and some intermittent problems caused by the non-stability of the input camera.

Some possible steps to make improvements in the model and continue with the project are exposed here:

- Improve the output performance in sharp turns. The actual model has problems to generate a decent predictions during the sharp turns of the circuit. This can be prevented by training the model with more sharp turns footage in the train data.
- Increase the number of classes to detect different object apart from the road.
- Introduction of LSTM network. During the first stages of the development of the model, one idea proposed was to implement and embedded LSTM Convolutional network right after the Encoder structure. This idea was based on the Robust Lane Detection from Continuous Driving Scenes paper [7] [13] .The idea behind this was to, instead predict the road using only one frame at a time, generate the prediction taking into account the previous generated predictions, saving the output of the last inner 2D Convolutional layer for a specific number of frames as a time series.
- Generate more data for the EgoLanes models. Increasing the existing dataset with FUSO footage including the same type of labeling as the TuSimple dataset could improve the performance of the models based on detecting the EgoLanes.

References

[1] Canny, J. A Computational Approach To Edge Detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **1986**, *PAMI-8*, 679–698.

[2] Zhou, S.; Jiang, Y.; Xi, J.; Gong, J.; Xiong, G.; Chen, H. In, **2010**, 59–64.

[3] Duda, R. O.; Hart, P. E. Use of the Hough transformation to detect lines and curves in pictures.**1972**.

[4] Ronneberger, O.; P.Fischer; Brox, T. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*; (available on arXiv:1505.04597 [cs.CV]), Springer: **2015**; *9351*, 234–241.

[5] Badrinarayan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Vehicular Technology* **2016**.

[6] Romera, E.; Álvarez, J. M.; Bergasa, L. M.; Arroyo, R. ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. *IEEE Transactions on Intelligent Transportation Systems* **2018**, *19*, 263–272.

[7] Zou, Q.; Jiang, H.; Dai, Q.; Yue, Y.; Chen, L.; Wang, Q. Robust lane detection from continuous driving scenes using deep neural networks. *IEEE Transactions on Vehicular Technology* **2019**.

[8] TuSimple TuSimple Github.

[9] Virgo, M. Lane Detection with Deep Learning.**2017**.

[10] Virgo, M. Michael Virgo Github.

[11] Wikipedia Jaccard index.

[12] Wikipedia F1 score.

[13] Q, Z.; H, J.; Q, D.; Y, Y.; L, C.; Q, W. Robust Lane Detection Github.