Machine Learning Engineer Nanodegree
Semantic Segmentation for self driving cars

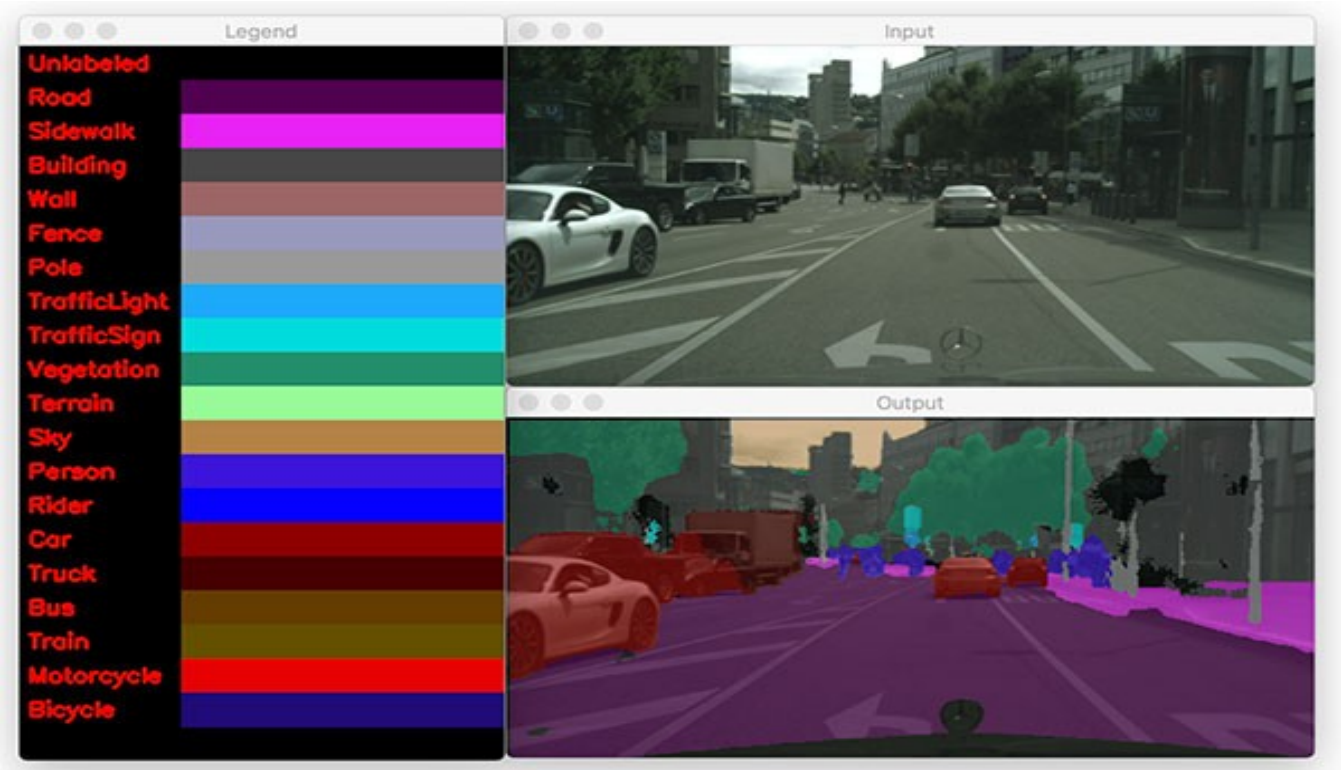
David Forino
January 23, 2019

Overview

Semantic segmentation is the task of clustering parts of images together which belong to the same object class. This type of algorithm has several use cases such as detecting road signs, cars, pedestrians, detecting tumors and much more.
Initial applications of computer vision required the identification of basic elements such as edges (lines and curves) or gradients. However understanding an image at pixel level came around only with the coining of full-pixel semantic segmentation.
Semantic segmentation is quite different and advanced compared to other image based tasks and don’t have to be confused,

- Image Classification identify what is present in the image.
- Object Recognition (and Detection) identify what is present in the image and where (via a Bounding Box).
- Semantic Segmentation identify what is present in the image and where (by finding all pixels that belong it).

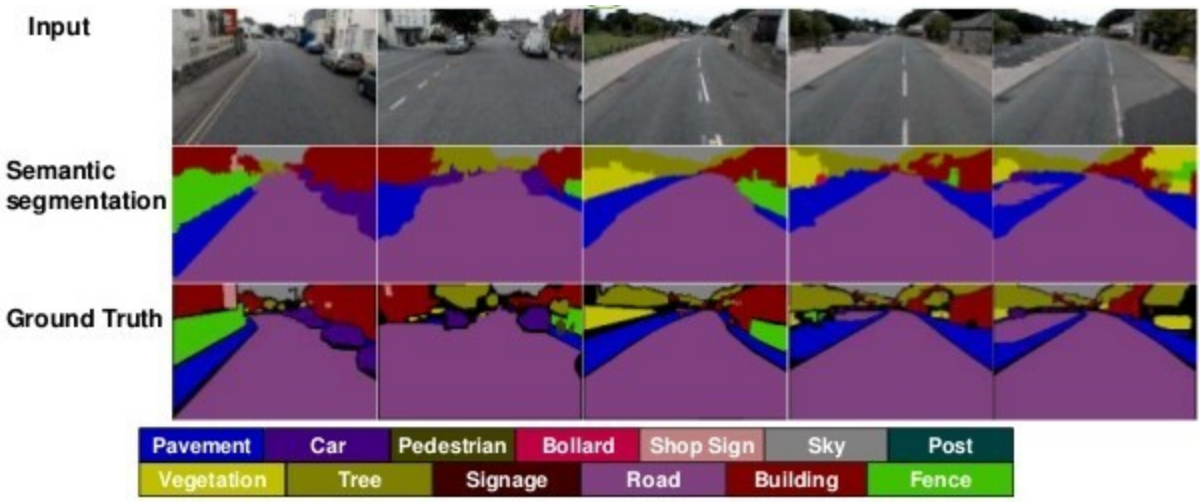
An example of Semantic Segmentation:



There are several techniques to perform semantic segmentation, for instance:
SEGNET, a Deep Convolutional Encoder-Decoder (research paper),
FCN, stands for Fully Convolutional Networks (research paper),
ENET, used for real time semantic segmentation (research paper) and more.
This is still a field of research and this paper you can see the progress in semantic image segmentation.

The Problem

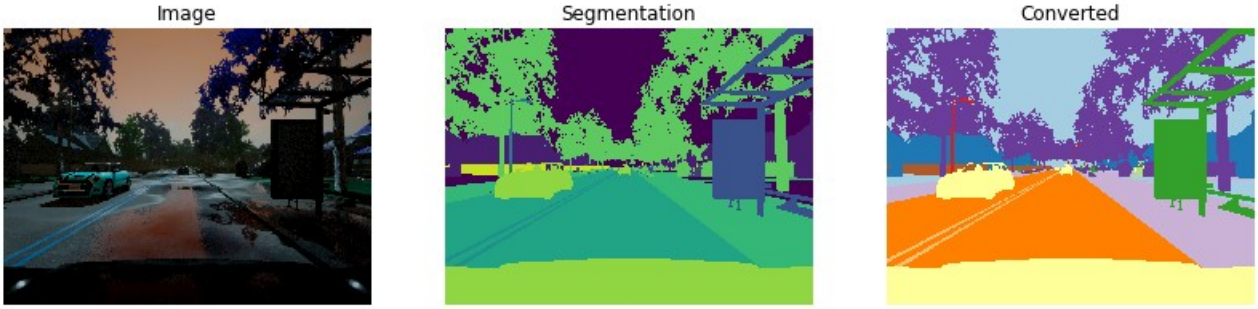
Autonomous driving is a complex robotic task that requires perception, planning and execution within constantly evolving environments. This task also needs to be performed with utmost precision, since safety is of paramount importance. Semantic Segmentation can provide information about free space on the roads, as well as detecting lane markings and traffic signs. This needs to be done many times per second in order to keep the car constantly updated with the actual state of the road.
In order to provide this information we need to correctly detect cars, roads, pedestrians, traffic signs and traffic lights in an image and their positions. In order to be able to train the model and evaluate it, is also necessary to have labels for all pixels corresponding to the related object, also called
“Ground Truth”.



This can be defined as a supervised classification problem.
Supervised, because we provide the Ground Truth and it is clearly a classification problem because every single pixel in the image needs to be classified as a part of the road, car, tree and so on.

Dataset

In order to solve this complicated problem I will use a Dataset from Kaggle. This dataset provides data images and labeled semantic segmentations captured via Carla self-driving car simulator.
The data was generated as part of the Lyft Udacity Challenge.
The data has 5 sets of 1000 images and corresponding labels and each set contains sets of RGB and the corresponding semantic segments.



To know more about how the simulator creates these images go to Sensor Camera. This simulator can provide us realistic scenarios for our model.

Solution Statement

To solve this problem I am going to implement the ENET model using Ternsorflow. There are many models to apply Semantic Segmentation but ENET seems the fastest because it has fewer parameters to train without loosing performance. It claims to be up to 18 times faster, requires 75 times less FLOPs and has 79 times less parameters. With this is mind, my aim is to be able to process images with at least 25 FPS speed.
For this project I will use a GTX 1050 Ti graphic card from NVIDIA, which is a good graphic card relatively inexpensive.
Of course with newer graphic cards we can achieve better performance in terms of speed.

Benchmark Model

I will then compare my solution against SEGNET model.
After recreating the same SEGNET neural network structure, I will also spend time on fine tuning the model, with also the possibility of creating a dedicated preprocess step for it.

Evaluation Metrics

Regarding the evaluation metric for the accuracy of the model I will use the mean IOU also known as “mean Intersection-Over-Union“, which is a common evaluation metric for semantic image segmentation. IOU is defined as follows:

IOU = true_positive / (true_positive + false_positive + false_negative)

To evaluate the speed I will simply measure the time to predict several images and divide it by the number of images. This process will be repeated several times in order to check the consistency.

Project Design

The first step in my project will be the visualization of an example image from the dataset. I do this in order to have a visual understanding of the differences between the real image and the Ground Truth. In order to be able to visualize the “Ground Truth” I have to cluster the red value of each pixel, for this I’ ll use “**numpy**” and for the visualization “**matplotlib**”.
Conversion example using python:

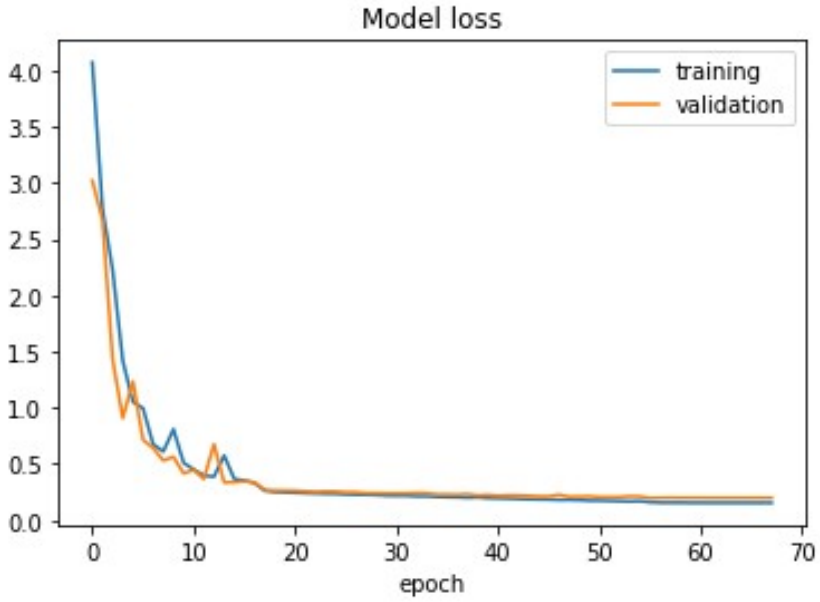
```
import numpy as np


def convert_segmentation(img):
    return np.apply_along_axis(lambda arr: max(arr), 2, img)
```

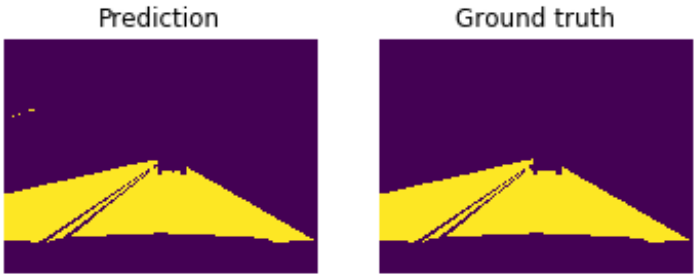
Image preprocessing will come afterward, “**openCV**” will play a very important role here, I might adjust color intensity, apply filters and try different techniques to see how the model reacts to these preprocessing steps.

As third step I’ll try to load all 5000 images in memory and split them into training, validation and test sets respectively 70%, 10% and 20%.
If there is not enough memory, I will split them into 2 chunks of 2500 images each.

In the fourth step I will define the ENET model using Tensorflow.
Here I will apply several training techniques to adjust some parameters of the model at runtime (an example would be the learning rate decay) in case it stops learning.

The fifth step is the model evaluation by first checking if there is any sign of overfitting or underfitting and in that case I will apply some techniques to overcome these problems, afterward there will be the image evaluation between prediction and ground truth.





Afterward, I will repeat step 4 and 5 using the SEGNET model instead, using the same procedures described above.

The final step will be apply the mean Intersection-Over-Union metric between the 2 models and the Ground Truth to evaluate mathematically the their precision and lastly, check the speed performance of both. This will require a lot of fine tuning to achieve the best performance, in case the speed of the model used for the solution is not appropriate I might reduce the image size, since a faster GPU would be too expensive.