AUTONOMOUS DRIVING VECHILE

1.Introduction

Autonomous driving relies heavily on large datasets of images and annotations to train deep learning models for tasks such as semantic segmentation, object detection, lane detection, and traffic sign recognition. The quality and structure of the dataset, along with appropriate preprocessing, are crucial to ensuring that the models generalize well. This document focuses on the preprocessing pipeline, including data loading functions, image preprocessing, annotation handling, and dataset splitting.

2.Dataset

2.1 Dataset Overview

The dataset used for this project consists solely of real world images captured in driving conditions. These images do not have accompanying annotations such as semantic segmentation masks, bounding boxes, lane markings, or traffic sign labels.
The preprocessing pipeline focuses on transforming the raw images into forms that can be used effectively by machine learning models, particularly for tasks like object detection and segmentation, once annotations are available or generated through other means.

Dataset Limitations

One notable limitation of the dataset is that it consists exclusively of images captured in daylight conditions As a result:

- **>** There are no images captured at night or in low light scenarios, which are critical for robust autonomous driving systems.
- **>** The dataset does not include images captured during rainy weather, limiting the model's ability to generalize to adverse weather conditions.

This lack of diverse conditions in the dataset may affect the model's performance when deployed in real-world environments where night driving or inclement weather is common.

3.Data Preprocessing

3.1 Data Loading Functions

The first step in data preprocessing involves loading images from the dataset. Since our dataset consists solely of images (without any annotations such as segmentation masks, bounding boxes, or lane markings), we will focus on handling the image files.
The images are typically stored in formats such as PNG or JPEG. To begin, we use the `glob` library to gather all image file paths from the specified directory.

Data loading steps include

- **>** Reading image files from the directory. In this case, we're using `glob.glob()` to retrieve all PNG files from the provided directory path (`image_direc`).
- **>** Once images are loaded, they can be pre-processed (e.g., resized to a standard resolution) before being fed into a machine learning model.
- **>** Efficient memory management is crucial when dealing with large datasets, so techniques like batch processing and image caching can be applied when needed.

```
[5] import glob
    import os
    image_direc = '/content/drive/MyDrive/LANES'
    image_paths = glob.glob(os.path.join(image_direc, "*.png"))
    print("successfully imported the images")

successfully imported the images
```

3.2 Image Preprocessing

Resizing : All images are resized to a fixed resolution of 1024x2048 to standardize the input for neural networks. This ensures that all images have the same dimensions, which is crucial for consistent model training and evaluation.

Focus on preprocessing images for semantic segmentation and object detection tasks. The following code snippet illustrates the application of a pre-trained DeepLabV3 model for semantic segmentation

code link: Segementation

This evaluation helps determine whether the dataset is compatible with segmentation tasks by analyzing how well the pre-trained model performs on the provided images, thereby assessing the quality and suitability of the dataset for semantic segmentation.

3.3 Image Transformation

Transformation of Gray-Scale Images: Using a separate image processing function, the dataset images that are initially in gray-scale format are transformed to resemble real-time images. This process enhances the visual quality of the images for subsequent tasks.

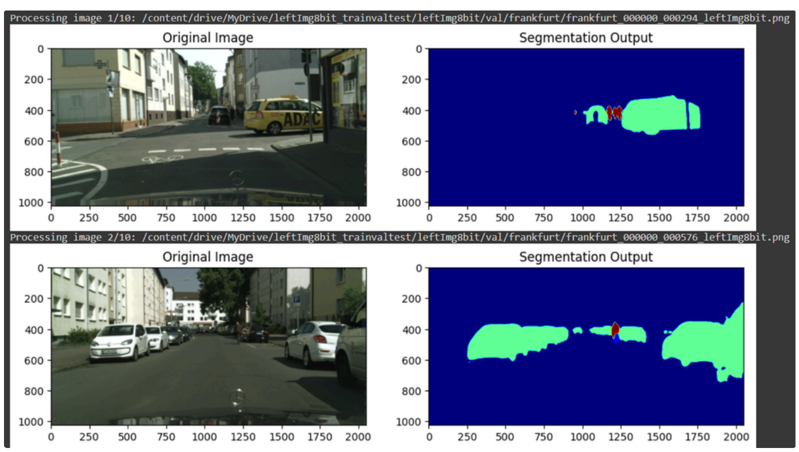
Color Gradation Matching: The transformation is achieved by matching the color gradation of a target image.

Focus on Lane Detection: This transformation is particularly important for tasks such as lane detection, where lane markings are typically represented in white. By enhancing the color representation of the images, the model's ability to accurately detect and segment lane markings is improved.

code link: Transformation

4.Results

The segmentation results highlight the model's effectiveness in distinguishing various elements within the scene. This is crucial for tasks such as object detection, lane detection, and understanding the driving environment. The model's performance on this sample indicates its potential applicability for real-time scenarios in autonomous driving systems.



The improved representation is especially beneficial for lane detection tasks, where lane markings are typically depicted in white. By ensuring that the colors are accurately represented, the model's ability to detect and segment lane markings effectively is enhanced, thereby improving the performance of autonomous driving systems.

5.Conclusion

After reviewing the dataset images, we found that they are suitable for training the segmentation model, as the pre-trained DeepLabV3 model is able to process and segment the images effectively. However, to make the model even better, we need more images taken in rainy and nighttime conditions. These different lighting and weather scenarios will help the model perform better in real-world situations. Additionally, transforming the existing gray-scale images into realistic colors has improved the quality of the dataset for detecting lanes. By adjusting the colors to match a target image, the lane markings are now easier to see. Moving forward, we will focus on adding more diverse images to the dataset to enhance the model's performance in various driving environments.

6.Future Objectives

- Train the model on various tasks, including semantic segmentation, object detection, lane detection, and traffic sign recognition. Each of these tasks plays a crucial role in enhancing the overall functionality of autonomous driving systems.
- A wider range of images will be included in the dataset, particularly those taken in different weather conditions and at various times of day. This will enhance the model's performance in real-world situations.
- Efforts will focus on making the model fast enough to analyze images in real time, which is crucial for driving applications. This may involve reducing the model's size to improve speed while maintaining accuracy.
- User feedback will be collected to understand how well the model performs in practical situations. This feedback will guide improvements over time.

7.References

Deeplabv3

How to convert a python numpy array to an RGB image with Opencv 2.4?