**TwinLiteNet: An Efficient and Lightweight Model for Drivable Area and Lane Segmentation in Self-Driving Cars**

**Team:**

1. Bal Narendra Sapa
2. Ajay Kumar Jagu

**Description:**

This project employs TwinLiteNet architecture as a starting point and it is modified to suit the requirements of road detection. The model will consist of one encoder and two decoders, with the encoder learning the representation of the input image and the decoders performing the segmentation of the road and non-road areas. We will incorporate Dual Attention Modules into the network to capture global dependencies in both spatial and channel dimensions, as suggested in the paper.

To train the model, we will use a dataset of road images with corresponding ground truth labels. We will use various loss functions, such as binary cross-entropy and dice loss and focal and Tversky loss, to optimize the model parameters. We will also use data augmentation techniques, such as random cropping and flipping, to increase the diversity of the training data and improve the model's generalization ability.

**Description of Dataset:**

The dataset used to train the model is BDD100k. With 100,000 frames and annotations for 10 tasks, it is a large dataset for autonomous driving. Due to its diversity in geography, environment, and weather conditions, the algorithm trained on the BDD100K dataset is robust enough to generalize to new settings. The BDD100K dataset is divided into three parts: a training set with 70,000 images, a validation set with 10,000 images, and a test set with 20,000 images.

**Data Source:**

Click [here](https://bdd-data.berkeley.edu/) to go the BDD100k Dataset site.

**Annotation Tools:**

The annotation tool that has been used in this project is “labelme”. The interface looks like this.

A road with trees in the background

Description automatically generated

Click [here](https://github.com/wkentaro/labelme) to go the official labelme repository.

This annotation tool is used to annotate the images for Drivable area segmentation and lane line detection.

**Samples:**

The images are annotated twice. One for Drivable area segmentation and another for lane line detection.

See the next page.

|  |  |  |
| --- | --- | --- |
| **Sample 1:**  Original image:  **A road with trees and grass**  Drivable area segmentation:  A red road with black background  Lane line detection:  A red object in the dark | | **Visualization:**  **A red road with trees in the background  Description automatically generated**  **A road with red lines  Description automatically generated** |
|  | |  |
| **Sample 2:**  Original image:  A road leading to mountains  Drivable Area Segmentation:  A red arrow in the sky  Lane line detection:  A red lines in a black background | **Visualization:**  **A red road leading to mountains**  **A road with red lines on it  Description automatically generated** | |

|  |  |
| --- | --- |
| **Sample 3:**  Original image:  A road with trees and bushes  Description automatically generated  Drivable Area Segmentation:  A red and black triangle with a black stripe  Description automatically generated  Lane line Detection:  A red line in the dark  Description automatically generated | **Visualization:**  **A red road with trees in the background**  **A road with red line  Description automatically generated** |

**Prompts used in Stable diffusion to generate images:**

The sample images are generated using stable diffusion. The model we have used is CompVis’s stable-diffusion-v1-4 model to generate the images. Link to the code is provided in the links section.

**Prompt used for sample 1:**

**A computer screen shot of a road

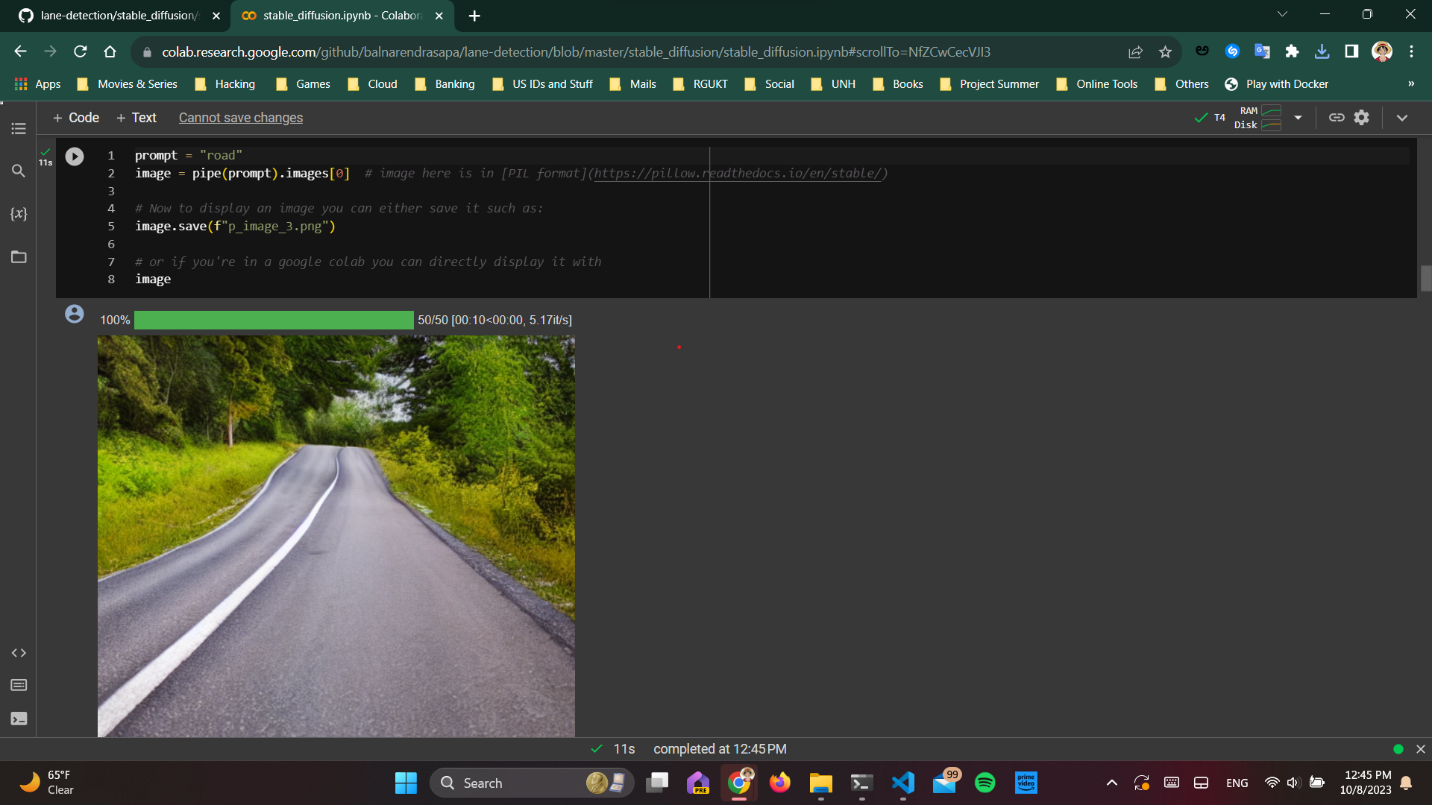
Description automatically generated**

**Prompt used for sample 2:**

**A screenshot of a computer

Description automatically generated**

**Prompt used for sample 3:**

****

**Links:**

1. **Link to the paper:**

Official arxiv link to the paper. Click [here.](https://arxiv.org/abs/2307.10705)

Official google scholar link to the paper. Click [here.](https://scholar.google.com/citations?view_op=view_citation&hl=vi&user=k7lUdFAAAAAJ&citation_for_view=k7lUdFAAAAAJ:d1gkVwhDpl0C)

1. **Link to the Online code used for the paper:**

The official link to the GitHub repository is provided here. Click [here.](https://github.com/chequanghuy/TwinLiteNet)

1. **Link to the Data-Source:**

BDD100k dataset was used here. Click [here.](https://bdd-data.berkeley.edu/)

1. **Link to the Annotation tool:**

Labelme annotation tool was used here to annotate the images of BDD100K dataset. Each image is annotated twice, one for drivable area and another for lane detection.

To go the official repository, click [here.](https://github.com/wkentaro/labelme)

1. **Link to the repository where we are presenting our work:**

Me and my teammate are pushing every change that we have worked on to GitHub. To go to the repository, click [here.](https://github.com/balnarendrasapa/road-detection)

1. **Link to the Google Colab that has code to generate images using stable diffusion:**

It is in the github repository provided in the 5th link. To go the file, click [here.](https://github.com/balnarendrasapa/road-detection/blob/master/stable_diffusion/stable_diffusion.ipynb)

After the file loads, click on the “Open in Google Colab” badge. Which opens the ipynb file in the google colab.