Building and Training Mini Network

December 15, 2023

0.1 Download the Repository

Repository Link

- This is our team's repository. This repository contains all the necessary code that we worked on and it also contains the dataset that we annotated.
- You do not need to do anything like uploading and adjusting the paths. Just run the cells sequentially.
- All the necessary commands are written in this notebook itself

[1]:
```
!git clone https://github.com/balnarendrasapa/road-detection.git
```

```
Cloning into 'road-detection'…
remote: Enumerating objects: 632, done.
remote: Counting objects: 100% (373/373), done.
remote: Compressing objects: 100% (262/262), done.
remote: Total 632 (delta 140), reused 223 (delta 84), pack-reused 259
Receiving objects: 100% (632/632), 212.18 MiB | 15.95 MiB/s, done.
Resolving deltas: 100% (233/233), done.
```

0.2 Install the Requirements

- Install all the python dependencies
- After Installing dependencies, Restart the runtime. If you do not restart the runtime, the python will throw "module not found error"

[2]:
```
!pip install -r road-detection/TwinLiteNet/requirements.txt
```

```
Collecting certifi==2023.7.22 (from -r road-
detection/TwinLiteNet/requirements.txt (line 1))
  Downloading certifi-2023.7.22-py3-none-any.whl (158 kB)
                             158.3/158.3
kB 3.0 MB/s eta 0:00:00
Requirement already satisfied: charset-normalizer==3.3.2 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 2)) (3.3.2)
Collecting colorama==0.4.6 (from -r road-detection/TwinLiteNet/requirements.txt
(line 3))
```

```
  Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Requirement already satisfied: contourpy==1.2.0 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 4)) (1.2.0)
Requirement already satisfied: cycler==0.12.1 in /usr/local/lib/python3.10/dist-
packages (from -r road-detection/TwinLiteNet/requirements.txt (line 5)) (0.12.1)
Collecting dnspython==2.4.2 (from -r road-detection/TwinLiteNet/requirements.txt
(line 6))
  Downloading dnspython-2.4.2-py3-none-any.whl (300 kB)
                             300.4/300.4

kB 11.3 MB/s eta 0:00:00
Collecting elephant==0.12.0 (from -r road-
detection/TwinLiteNet/requirements.txt (line 7))
  Downloading
elephant-0.12.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.3
MB)
                             1.3/1.3 MB
8.1 MB/s eta 0:00:00
Requirement already satisfied: filelock==3.13.1 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 8)) (3.13.1)
Collecting fonttools==4.44.0 (from -r road-
detection/TwinLiteNet/requirements.txt (line 9))
  Downloading
fonttools-4.44.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.5
MB)
                             4.5/4.5 MB
38.5 MB/s eta 0:00:00
Collecting fsspec==2023.10.0 (from -r road-
detection/TwinLiteNet/requirements.txt (line 10))
  Downloading fsspec-2023.10.0-py3-none-any.whl (166 kB)
                             166.4/166.4

kB 24.0 MB/s eta 0:00:00
Collecting idna==3.4 (from -r road-detection/TwinLiteNet/requirements.txt
(line 11))
  Downloading idna-3.4-py3-none-any.whl (61 kB)
                             61.5/61.5 kB
9.1 MB/s eta 0:00:00
Requirement already satisfied: Jinja2==3.1.2 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 12)) (3.1.2)
Collecting joblib==1.2.0 (from -r road-detection/TwinLiteNet/requirements.txt
(line 13))
  Downloading joblib-1.2.0-py3-none-any.whl (297 kB)
                             298.0/298.0

kB 36.4 MB/s eta 0:00:00
```

```
Requirement already satisfied: kiwisolver==1.4.5 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 14)) (1.4.5)
Requirement already satisfied: MarkupSafe==2.1.3 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 15)) (2.1.3)
Requirement already satisfied: matplotlib==3.7.1 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 16)) (3.7.1)
Requirement already satisfied: mpmath==1.3.0 in /usr/local/lib/python3.10/dist-
packages (from -r road-detection/TwinLiteNet/requirements.txt (line 17)) (1.3.0)
Collecting neo==0.12.0 (from -r road-detection/TwinLiteNet/requirements.txt
(line 18))
  Downloading neo-0.12.0-py3-none-any.whl (586 kB)
                             586.9/586.9
kB 37.3 MB/s eta 0:00:00
Requirement already satisfied: networkx==3.2.1 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 19)) (3.2.1)
Collecting numpy==1.24.3 (from -r road-detection/TwinLiteNet/requirements.txt
(line 20))
  Downloading
numpy-1.24.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.3
MB)
                             17.3/17.3 MB
37.5 MB/s eta 0:00:00
Collecting opencv-python==4.7.0.72 (from -r road-
detection/TwinLiteNet/requirements.txt (line 21))
  Downloading
opencv_python-4.7.0.72-cp37-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(61.8 MB)
                             61.8/61.8 MB
8.4 MB/s eta 0:00:00
Requirement already satisfied: packaging==23.2 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 22)) (23.2)
Collecting Pillow==9.5.0 (from -r road-detection/TwinLiteNet/requirements.txt
(line 23))
  Downloading Pillow-9.5.0-cp310-cp310-manylinux_2_28_x86_64.whl (3.4 MB)
                             3.4/3.4 MB
55.5 MB/s eta 0:00:00
Requirement already satisfied: pyparsing==3.1.1 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 24)) (3.1.1)
Requirement already satisfied: python-dateutil==2.8.2 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 25)) (2.8.2)
```

```
Collecting python-etcd==0.4.5 (from -r road-
detection/TwinLiteNet/requirements.txt (line 26))
  Downloading python-etcd-0.4.5.tar.gz (37 kB)
  Preparing metadata (setup.py) … done
Requirement already satisfied: PyYAML==6.0.1 in /usr/local/lib/python3.10/dist-
packages (from -r road-detection/TwinLiteNet/requirements.txt (line 27)) (6.0.1)
Collecting quantities==0.14.1 (from -r road-
detection/TwinLiteNet/requirements.txt (line 28))
  Downloading quantities-0.14.1-py3-none-any.whl (87 kB)
                             87.9/87.9 kB
12.0 MB/s eta 0:00:00
Requirement already satisfied: requests==2.31.0 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 29)) (2.31.0)
Collecting scikit-learn==1.3.2 (from -r road-
detection/TwinLiteNet/requirements.txt (line 30))
  Downloading
scikit_learn-1.3.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(10.8 MB)
                             10.8/10.8 MB
95.5 MB/s eta 0:00:00
Collecting scipy==1.10.1 (from -r road-
detection/TwinLiteNet/requirements.txt (line 31))
  Downloading
scipy-1.10.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (34.4
MB)
                             34.4/34.4 MB
17.1 MB/s eta 0:00:00
Requirement already satisfied: six==1.16.0 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 32)) (1.16.0)
Requirement already satisfied: sympy==1.12 in /usr/local/lib/python3.10/dist-
packages (from -r road-detection/TwinLiteNet/requirements.txt (line 33)) (1.12)
Requirement already satisfied: threadpoolctl==3.2.0 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 34)) (3.2.0)
Requirement already satisfied: torch==2.1.0 in /usr/local/lib/python3.10/dist-
packages (from -r road-detection/TwinLiteNet/requirements.txt (line 35))
(2.1.0+cu121)
Requirement already satisfied: torchdata==0.7.0 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 36)) (0.7.0)
Collecting torchelastic==0.2.2 (from -r road-
detection/TwinLiteNet/requirements.txt (line 37))
  Downloading torchelastic-0.2.2-py3-none-any.whl (111 kB)
                             111.5/111.5
kB 14.8 MB/s eta 0:00:00
```

```
Requirement already satisfied: torchtext==0.16.0 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 38)) (0.16.0)
Requirement already satisfied: torchvision==0.16.0 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 39)) (0.16.0+cu121)
Requirement already satisfied: tqdm==4.66.1 in /usr/local/lib/python3.10/dist-
packages (from -r road-detection/TwinLiteNet/requirements.txt (line 40))
(4.66.1)
Collecting typing_extensions==4.8.0 (from -r road-
detection/TwinLiteNet/requirements.txt (line 41))
  Downloading typing_extensions-4.8.0-py3-none-any.whl (31 kB)
Requirement already satisfied: urllib3==2.0.7 in /usr/local/lib/python3.10/dist-
packages (from -r road-detection/TwinLiteNet/requirements.txt (line 42)) (2.0.7)
Requirement already satisfied: webcolors==1.13 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 43)) (1.13)
Collecting yacs==0.1.8 (from -r road-detection/TwinLiteNet/requirements.txt
(line 44))
  Downloading yacs-0.1.8-py3-none-any.whl (14 kB)
Collecting zipp==3.15.0 (from -r road-detection/TwinLiteNet/requirements.txt
(line 45))
  Downloading zipp-3.15.0-py3-none-any.whl (6.8 kB)
Requirement already satisfied: triton==2.1.0 in /usr/local/lib/python3.10/dist-
packages (from torch==2.1.0->-r road-detection/TwinLiteNet/requirements.txt
(line 35)) (2.1.0)
Building wheels for collected packages: python-etcd
  Building wheel for python-etcd (setup.py) … done
  Created wheel for python-etcd: filename=python_etcd-0.4.5-py3-none-any.whl
size=38481
sha256=4b72c1f1faa05dee1f634adb3b34fda41f49c617a915022b2bf7f907cc899689
  Stored in directory: /root/.cache/pip/wheels/93/5f/1b/056db07a0ab1c0b7efe17592
8d2a10b614e0e00d7bab0b6496
Successfully built python-etcd
Installing collected packages: zipp, yacs, typing_extensions, Pillow, numpy,
joblib, idna, fsspec, fonttools, dnspython, colorama, certifi, scipy,
quantities, python-etcd, opencv-python, torchelastic, scikit-learn, neo,
elephant
  Attempting uninstall: zipp
    Found existing installation: zipp 3.17.0
    Uninstalling zipp-3.17.0:
      Successfully uninstalled zipp-3.17.0
  Attempting uninstall: typing_extensions
    Found existing installation: typing_extensions 4.5.0
    Uninstalling typing_extensions-4.5.0:
      Successfully uninstalled typing_extensions-4.5.0
  Attempting uninstall: Pillow
    Found existing installation: Pillow 9.4.0
```

```
  Uninstalling Pillow-9.4.0:
    Successfully uninstalled Pillow-9.4.0
Attempting uninstall: numpy
  Found existing installation: numpy 1.23.5
  Uninstalling numpy-1.23.5:
    Successfully uninstalled numpy-1.23.5
Attempting uninstall: joblib
  Found existing installation: joblib 1.3.2
  Uninstalling joblib-1.3.2:
    Successfully uninstalled joblib-1.3.2
Attempting uninstall: idna
  Found existing installation: idna 3.6
  Uninstalling idna-3.6:
    Successfully uninstalled idna-3.6
Attempting uninstall: fsspec
  Found existing installation: fsspec 2023.6.0
  Uninstalling fsspec-2023.6.0:
    Successfully uninstalled fsspec-2023.6.0
Attempting uninstall: fonttools
  Found existing installation: fonttools 4.46.0
  Uninstalling fonttools-4.46.0:
    Successfully uninstalled fonttools-4.46.0
Attempting uninstall: certifi
  Found existing installation: certifi 2023.11.17
  Uninstalling certifi-2023.11.17:
    Successfully uninstalled certifi-2023.11.17
Attempting uninstall: scipy
  Found existing installation: scipy 1.11.4
  Uninstalling scipy-1.11.4:
    Successfully uninstalled scipy-1.11.4
Attempting uninstall: opencv-python
  Found existing installation: opencv-python 4.8.0.76
  Uninstalling opencv-python-4.8.0.76:
    Successfully uninstalled opencv-python-4.8.0.76
Attempting uninstall: scikit-learn
  Found existing installation: scikit-learn 1.2.2
  Uninstalling scikit-learn-1.2.2:
    Successfully uninstalled scikit-learn-1.2.2
```

```
ERROR: pip's dependency resolver does not currently take into account all
the packages that are installed. This behaviour is the source of the following
dependency conflicts.
lida 0.0.10 requires fastapi, which is not installed.
lida 0.0.10 requires kaleido, which is not installed.
lida 0.0.10 requires python-multipart, which is not installed.
lida 0.0.10 requires uvicorn, which is not installed.
gcsfs 2023.6.0 requires fsspec==2023.6.0, but you have fsspec 2023.10.0 which is
incompatible.
tensorflow-probability 0.22.0 requires typing-extensions<4.6.0, but you have
typing-extensions 4.8.0 which is incompatible.
Successfully installed Pillow-9.5.0 certifi-2023.7.22 colorama-0.4.6
dnspython-2.4.2 elephant-0.12.0 fonttools-4.44.0 fsspec-2023.10.0 idna-3.4
joblib-1.2.0 neo-0.12.0 numpy-1.24.3 opencv-python-4.7.0.72 python-etcd-0.4.5
quantities-0.14.1 scikit-learn-1.3.2 scipy-1.10.1 torchelastic-0.2.2
typing_extensions-4.8.0 yacs-0.1.8 zipp-3.15.0
```

0.3 Copy Dataset from Repository

- Our repository contains dataset.zip in datasets folder in the repository. copy that zip file to root

[1]:
```
!cp road-detection/datasets/dataset.zip ./
```

0.4 Unzip the file

[2]:
```
!unzip dataset.zip
```

```
Archive:  dataset.zip
   creating: dataset/test/
   creating: dataset/test/images/
  inflating: dataset/test/images/road_image_160.png
  inflating: dataset/test/images/road_image_161.png
  inflating: dataset/test/images/road_image_162.png
  inflating: dataset/test/images/road_image_163.png
  inflating: dataset/test/images/road_image_164.png
  inflating: dataset/test/images/road_image_165.png
  inflating: dataset/test/images/road_image_166.png
  inflating: dataset/test/images/road_image_167.png
  inflating: dataset/test/images/road_image_168.png
  inflating: dataset/test/images/road_image_169.png
  inflating: dataset/test/images/road_image_170.png
```

```
inflating: dataset/test/images/road_image_171.png
inflating: dataset/test/images/road_image_172.png
inflating: dataset/test/images/road_image_173.png
inflating: dataset/test/images/road_image_174.png
inflating: dataset/test/images/road_image_175.png
inflating: dataset/test/images/road_image_176.png
inflating: dataset/test/images/road_image_177.png
inflating: dataset/test/images/road_image_178.png
inflating: dataset/test/images/road_image_179.png
 creating: dataset/test/lane/
inflating: dataset/test/lane/road_image_160.png
inflating: dataset/test/lane/road_image_161.png
inflating: dataset/test/lane/road_image_162.png
inflating: dataset/test/lane/road_image_163.png
inflating: dataset/test/lane/road_image_164.png
inflating: dataset/test/lane/road_image_165.png
inflating: dataset/test/lane/road_image_166.png
inflating: dataset/test/lane/road_image_167.png
inflating: dataset/test/lane/road_image_168.png
inflating: dataset/test/lane/road_image_169.png
inflating: dataset/test/lane/road_image_170.png
inflating: dataset/test/lane/road_image_171.png
inflating: dataset/test/lane/road_image_172.png
inflating: dataset/test/lane/road_image_173.png
inflating: dataset/test/lane/road_image_174.png
inflating: dataset/test/lane/road_image_175.png
inflating: dataset/test/lane/road_image_176.png
inflating: dataset/test/lane/road_image_177.png
inflating: dataset/test/lane/road_image_178.png
inflating: dataset/test/lane/road_image_179.png
 creating: dataset/test/segments/
inflating: dataset/test/segments/road_image_160.png
inflating: dataset/test/segments/road_image_161.png
inflating: dataset/test/segments/road_image_162.png
inflating: dataset/test/segments/road_image_163.png
inflating: dataset/test/segments/road_image_164.png
inflating: dataset/test/segments/road_image_165.png
inflating: dataset/test/segments/road_image_166.png
inflating: dataset/test/segments/road_image_167.png
inflating: dataset/test/segments/road_image_168.png
inflating: dataset/test/segments/road_image_169.png
inflating: dataset/test/segments/road_image_170.png
inflating: dataset/test/segments/road_image_171.png
inflating: dataset/test/segments/road_image_172.png
inflating: dataset/test/segments/road_image_173.png
inflating: dataset/test/segments/road_image_174.png
inflating: dataset/test/segments/road_image_175.png
inflating: dataset/test/segments/road_image_176.png
```

```
inflating: dataset/test/segments/road_image_177.png
inflating: dataset/test/segments/road_image_178.png
inflating: dataset/test/segments/road_image_179.png
 creating: dataset/train/
 creating: dataset/train/images/
inflating: dataset/train/images/road_image_0.png
inflating: dataset/train/images/road_image_1.png
inflating: dataset/train/images/road_image_10.png
inflating: dataset/train/images/road_image_100.png
inflating: dataset/train/images/road_image_101.png
inflating: dataset/train/images/road_image_102.png
inflating: dataset/train/images/road_image_103.png
inflating: dataset/train/images/road_image_104.png
inflating: dataset/train/images/road_image_105.png
inflating: dataset/train/images/road_image_106.png
inflating: dataset/train/images/road_image_107.png
inflating: dataset/train/images/road_image_108.png
inflating: dataset/train/images/road_image_109.png
inflating: dataset/train/images/road_image_11.png
inflating: dataset/train/images/road_image_110.png
inflating: dataset/train/images/road_image_111.png
inflating: dataset/train/images/road_image_112.png
inflating: dataset/train/images/road_image_113.png
inflating: dataset/train/images/road_image_114.png
inflating: dataset/train/images/road_image_115.png
inflating: dataset/train/images/road_image_116.png
inflating: dataset/train/images/road_image_117.png
inflating: dataset/train/images/road_image_118.png
inflating: dataset/train/images/road_image_119.png
inflating: dataset/train/images/road_image_12.png
inflating: dataset/train/images/road_image_120.png
inflating: dataset/train/images/road_image_121.png
inflating: dataset/train/images/road_image_122.png
inflating: dataset/train/images/road_image_123.png
inflating: dataset/train/images/road_image_124.png
inflating: dataset/train/images/road_image_125.png
inflating: dataset/train/images/road_image_126.png
inflating: dataset/train/images/road_image_127.png
inflating: dataset/train/images/road_image_128.png
inflating: dataset/train/images/road_image_129.png
inflating: dataset/train/images/road_image_13.png
inflating: dataset/train/images/road_image_130.png
inflating: dataset/train/images/road_image_131.png
inflating: dataset/train/images/road_image_132.png
inflating: dataset/train/images/road_image_133.png
inflating: dataset/train/images/road_image_134.png
inflating: dataset/train/images/road_image_135.png
inflating: dataset/train/images/road_image_136.png
```

```
inflating: dataset/train/images/road_image_137.png
inflating: dataset/train/images/road_image_138.png
inflating: dataset/train/images/road_image_139.png
inflating: dataset/train/images/road_image_14.png
inflating: dataset/train/images/road_image_140.png
inflating: dataset/train/images/road_image_141.png
inflating: dataset/train/images/road_image_142.png
inflating: dataset/train/images/road_image_143.png
inflating: dataset/train/images/road_image_144.png
inflating: dataset/train/images/road_image_145.png
inflating: dataset/train/images/road_image_146.png
inflating: dataset/train/images/road_image_147.png
inflating: dataset/train/images/road_image_148.png
inflating: dataset/train/images/road_image_149.png
inflating: dataset/train/images/road_image_15.png
inflating: dataset/train/images/road_image_150.png
inflating: dataset/train/images/road_image_151.png
inflating: dataset/train/images/road_image_152.png
inflating: dataset/train/images/road_image_153.png
inflating: dataset/train/images/road_image_154.png
inflating: dataset/train/images/road_image_155.png
inflating: dataset/train/images/road_image_156.png
inflating: dataset/train/images/road_image_157.png
inflating: dataset/train/images/road_image_158.png
inflating: dataset/train/images/road_image_159.png
inflating: dataset/train/images/road_image_16.png
inflating: dataset/train/images/road_image_17.png
inflating: dataset/train/images/road_image_18.png
inflating: dataset/train/images/road_image_19.png
inflating: dataset/train/images/road_image_2.png
inflating: dataset/train/images/road_image_20.png
inflating: dataset/train/images/road_image_21.png
inflating: dataset/train/images/road_image_22.png
inflating: dataset/train/images/road_image_23.png
inflating: dataset/train/images/road_image_24.png
inflating: dataset/train/images/road_image_25.png
inflating: dataset/train/images/road_image_26.png
inflating: dataset/train/images/road_image_27.png
inflating: dataset/train/images/road_image_28.png
inflating: dataset/train/images/road_image_29.png
inflating: dataset/train/images/road_image_3.png
inflating: dataset/train/images/road_image_30.png
inflating: dataset/train/images/road_image_31.png
inflating: dataset/train/images/road_image_32.png
inflating: dataset/train/images/road_image_33.png
inflating: dataset/train/images/road_image_34.png
inflating: dataset/train/images/road_image_35.png
inflating: dataset/train/images/road_image_36.png
```

```
inflating: dataset/train/images/road_image_37.png
inflating: dataset/train/images/road_image_38.png
inflating: dataset/train/images/road_image_39.png
inflating: dataset/train/images/road_image_4.png
inflating: dataset/train/images/road_image_40.png
inflating: dataset/train/images/road_image_41.png
inflating: dataset/train/images/road_image_42.png
inflating: dataset/train/images/road_image_43.png
inflating: dataset/train/images/road_image_44.png
inflating: dataset/train/images/road_image_45.png
inflating: dataset/train/images/road_image_46.png
inflating: dataset/train/images/road_image_47.png
inflating: dataset/train/images/road_image_48.png
inflating: dataset/train/images/road_image_49.png
inflating: dataset/train/images/road_image_5.png
inflating: dataset/train/images/road_image_50.png
inflating: dataset/train/images/road_image_51.png
inflating: dataset/train/images/road_image_52.png
inflating: dataset/train/images/road_image_53.png
inflating: dataset/train/images/road_image_54.png
inflating: dataset/train/images/road_image_55.png
inflating: dataset/train/images/road_image_56.png
inflating: dataset/train/images/road_image_57.png
inflating: dataset/train/images/road_image_58.png
inflating: dataset/train/images/road_image_59.png
inflating: dataset/train/images/road_image_6.png
inflating: dataset/train/images/road_image_60.png
inflating: dataset/train/images/road_image_61.png
inflating: dataset/train/images/road_image_62.png
inflating: dataset/train/images/road_image_63.png
inflating: dataset/train/images/road_image_64.png
inflating: dataset/train/images/road_image_65.png
inflating: dataset/train/images/road_image_66.png
inflating: dataset/train/images/road_image_67.png
inflating: dataset/train/images/road_image_68.png
inflating: dataset/train/images/road_image_69.png
inflating: dataset/train/images/road_image_7.png
inflating: dataset/train/images/road_image_70.png
inflating: dataset/train/images/road_image_71.png
inflating: dataset/train/images/road_image_72.png
inflating: dataset/train/images/road_image_73.png
inflating: dataset/train/images/road_image_74.png
inflating: dataset/train/images/road_image_75.png
inflating: dataset/train/images/road_image_76.png
inflating: dataset/train/images/road_image_77.png
inflating: dataset/train/images/road_image_78.png
inflating: dataset/train/images/road_image_79.png
inflating: dataset/train/images/road_image_8.png
```

```
inflating: dataset/train/images/road_image_80.png
inflating: dataset/train/images/road_image_81.png
inflating: dataset/train/images/road_image_82.png
inflating: dataset/train/images/road_image_83.png
inflating: dataset/train/images/road_image_84.png
inflating: dataset/train/images/road_image_85.png
inflating: dataset/train/images/road_image_86.png
inflating: dataset/train/images/road_image_87.png
inflating: dataset/train/images/road_image_88.png
inflating: dataset/train/images/road_image_89.png
inflating: dataset/train/images/road_image_9.png
inflating: dataset/train/images/road_image_90.png
inflating: dataset/train/images/road_image_91.png
inflating: dataset/train/images/road_image_92.png
inflating: dataset/train/images/road_image_93.png
inflating: dataset/train/images/road_image_94.png
inflating: dataset/train/images/road_image_95.png
inflating: dataset/train/images/road_image_96.png
inflating: dataset/train/images/road_image_97.png
inflating: dataset/train/images/road_image_98.png
inflating: dataset/train/images/road_image_99.png
 creating: dataset/train/lane/
inflating: dataset/train/lane/road_image_0.png
inflating: dataset/train/lane/road_image_1.png
inflating: dataset/train/lane/road_image_10.png
inflating: dataset/train/lane/road_image_100.png
inflating: dataset/train/lane/road_image_101.png
inflating: dataset/train/lane/road_image_102.png
inflating: dataset/train/lane/road_image_103.png
inflating: dataset/train/lane/road_image_104.png
inflating: dataset/train/lane/road_image_105.png
inflating: dataset/train/lane/road_image_106.png
inflating: dataset/train/lane/road_image_107.png
inflating: dataset/train/lane/road_image_108.png
inflating: dataset/train/lane/road_image_109.png
inflating: dataset/train/lane/road_image_11.png
inflating: dataset/train/lane/road_image_110.png
inflating: dataset/train/lane/road_image_111.png
inflating: dataset/train/lane/road_image_112.png
inflating: dataset/train/lane/road_image_113.png
inflating: dataset/train/lane/road_image_114.png
inflating: dataset/train/lane/road_image_115.png
inflating: dataset/train/lane/road_image_116.png
inflating: dataset/train/lane/road_image_117.png
inflating: dataset/train/lane/road_image_118.png
inflating: dataset/train/lane/road_image_119.png
inflating: dataset/train/lane/road_image_12.png
inflating: dataset/train/lane/road_image_120.png
```

```
inflating: dataset/train/lane/road_image_121.png
inflating: dataset/train/lane/road_image_122.png
inflating: dataset/train/lane/road_image_123.png
inflating: dataset/train/lane/road_image_124.png
inflating: dataset/train/lane/road_image_125.png
inflating: dataset/train/lane/road_image_126.png
inflating: dataset/train/lane/road_image_127.png
inflating: dataset/train/lane/road_image_128.png
inflating: dataset/train/lane/road_image_129.png
inflating: dataset/train/lane/road_image_13.png
inflating: dataset/train/lane/road_image_130.png
inflating: dataset/train/lane/road_image_131.png
inflating: dataset/train/lane/road_image_132.png
inflating: dataset/train/lane/road_image_133.png
inflating: dataset/train/lane/road_image_134.png
inflating: dataset/train/lane/road_image_135.png
inflating: dataset/train/lane/road_image_136.png
inflating: dataset/train/lane/road_image_137.png
inflating: dataset/train/lane/road_image_138.png
inflating: dataset/train/lane/road_image_139.png
inflating: dataset/train/lane/road_image_14.png
inflating: dataset/train/lane/road_image_140.png
inflating: dataset/train/lane/road_image_141.png
inflating: dataset/train/lane/road_image_142.png
inflating: dataset/train/lane/road_image_143.png
inflating: dataset/train/lane/road_image_144.png
inflating: dataset/train/lane/road_image_145.png
inflating: dataset/train/lane/road_image_146.png
inflating: dataset/train/lane/road_image_147.png
inflating: dataset/train/lane/road_image_148.png
inflating: dataset/train/lane/road_image_149.png
inflating: dataset/train/lane/road_image_15.png
inflating: dataset/train/lane/road_image_150.png
inflating: dataset/train/lane/road_image_151.png
inflating: dataset/train/lane/road_image_152.png
inflating: dataset/train/lane/road_image_153.png
inflating: dataset/train/lane/road_image_154.png
inflating: dataset/train/lane/road_image_155.png
inflating: dataset/train/lane/road_image_156.png
inflating: dataset/train/lane/road_image_157.png
inflating: dataset/train/lane/road_image_158.png
inflating: dataset/train/lane/road_image_159.png
inflating: dataset/train/lane/road_image_16.png
inflating: dataset/train/lane/road_image_17.png
inflating: dataset/train/lane/road_image_18.png
inflating: dataset/train/lane/road_image_19.png
inflating: dataset/train/lane/road_image_2.png
inflating: dataset/train/lane/road_image_20.png
```

```
inflating: dataset/train/lane/road_image_21.png
inflating: dataset/train/lane/road_image_22.png
inflating: dataset/train/lane/road_image_23.png
inflating: dataset/train/lane/road_image_24.png
inflating: dataset/train/lane/road_image_25.png
inflating: dataset/train/lane/road_image_26.png
inflating: dataset/train/lane/road_image_27.png
inflating: dataset/train/lane/road_image_28.png
inflating: dataset/train/lane/road_image_29.png
inflating: dataset/train/lane/road_image_3.png
inflating: dataset/train/lane/road_image_30.png
inflating: dataset/train/lane/road_image_31.png
inflating: dataset/train/lane/road_image_32.png
inflating: dataset/train/lane/road_image_33.png
inflating: dataset/train/lane/road_image_34.png
inflating: dataset/train/lane/road_image_35.png
inflating: dataset/train/lane/road_image_36.png
inflating: dataset/train/lane/road_image_37.png
inflating: dataset/train/lane/road_image_38.png
inflating: dataset/train/lane/road_image_39.png
inflating: dataset/train/lane/road_image_4.png
inflating: dataset/train/lane/road_image_40.png
inflating: dataset/train/lane/road_image_41.png
inflating: dataset/train/lane/road_image_42.png
inflating: dataset/train/lane/road_image_43.png
inflating: dataset/train/lane/road_image_44.png
inflating: dataset/train/lane/road_image_45.png
inflating: dataset/train/lane/road_image_46.png
inflating: dataset/train/lane/road_image_47.png
inflating: dataset/train/lane/road_image_48.png
inflating: dataset/train/lane/road_image_49.png
inflating: dataset/train/lane/road_image_5.png
inflating: dataset/train/lane/road_image_50.png
inflating: dataset/train/lane/road_image_51.png
inflating: dataset/train/lane/road_image_52.png
inflating: dataset/train/lane/road_image_53.png
inflating: dataset/train/lane/road_image_54.png
inflating: dataset/train/lane/road_image_55.png
inflating: dataset/train/lane/road_image_56.png
inflating: dataset/train/lane/road_image_57.png
inflating: dataset/train/lane/road_image_58.png
inflating: dataset/train/lane/road_image_59.png
inflating: dataset/train/lane/road_image_6.png
inflating: dataset/train/lane/road_image_60.png
inflating: dataset/train/lane/road_image_61.png
inflating: dataset/train/lane/road_image_62.png
inflating: dataset/train/lane/road_image_63.png
inflating: dataset/train/lane/road_image_64.png
```

```
inflating: dataset/train/lane/road_image_65.png
inflating: dataset/train/lane/road_image_66.png
inflating: dataset/train/lane/road_image_67.png
inflating: dataset/train/lane/road_image_68.png
inflating: dataset/train/lane/road_image_69.png
inflating: dataset/train/lane/road_image_7.png
inflating: dataset/train/lane/road_image_70.png
inflating: dataset/train/lane/road_image_71.png
inflating: dataset/train/lane/road_image_72.png
inflating: dataset/train/lane/road_image_73.png
inflating: dataset/train/lane/road_image_74.png
inflating: dataset/train/lane/road_image_75.png
inflating: dataset/train/lane/road_image_76.png
inflating: dataset/train/lane/road_image_77.png
inflating: dataset/train/lane/road_image_78.png
inflating: dataset/train/lane/road_image_79.png
inflating: dataset/train/lane/road_image_8.png
inflating: dataset/train/lane/road_image_80.png
inflating: dataset/train/lane/road_image_81.png
inflating: dataset/train/lane/road_image_82.png
inflating: dataset/train/lane/road_image_83.png
inflating: dataset/train/lane/road_image_84.png
inflating: dataset/train/lane/road_image_85.png
inflating: dataset/train/lane/road_image_86.png
inflating: dataset/train/lane/road_image_87.png
inflating: dataset/train/lane/road_image_88.png
inflating: dataset/train/lane/road_image_89.png
inflating: dataset/train/lane/road_image_9.png
inflating: dataset/train/lane/road_image_90.png
inflating: dataset/train/lane/road_image_91.png
inflating: dataset/train/lane/road_image_92.png
inflating: dataset/train/lane/road_image_93.png
inflating: dataset/train/lane/road_image_94.png
inflating: dataset/train/lane/road_image_95.png
inflating: dataset/train/lane/road_image_96.png
inflating: dataset/train/lane/road_image_97.png
inflating: dataset/train/lane/road_image_98.png
inflating: dataset/train/lane/road_image_99.png
 creating: dataset/train/segments/
inflating: dataset/train/segments/road_image_0.png
inflating: dataset/train/segments/road_image_1.png
inflating: dataset/train/segments/road_image_10.png
inflating: dataset/train/segments/road_image_100.png
inflating: dataset/train/segments/road_image_101.png
inflating: dataset/train/segments/road_image_102.png
inflating: dataset/train/segments/road_image_103.png
inflating: dataset/train/segments/road_image_104.png
inflating: dataset/train/segments/road_image_105.png
```

```
inflating: dataset/train/segments/road_image_106.png
inflating: dataset/train/segments/road_image_107.png
inflating: dataset/train/segments/road_image_108.png
inflating: dataset/train/segments/road_image_109.png
inflating: dataset/train/segments/road_image_11.png
inflating: dataset/train/segments/road_image_110.png
inflating: dataset/train/segments/road_image_111.png
inflating: dataset/train/segments/road_image_112.png
inflating: dataset/train/segments/road_image_113.png
inflating: dataset/train/segments/road_image_114.png
inflating: dataset/train/segments/road_image_115.png
inflating: dataset/train/segments/road_image_116.png
inflating: dataset/train/segments/road_image_117.png
inflating: dataset/train/segments/road_image_118.png
inflating: dataset/train/segments/road_image_119.png
inflating: dataset/train/segments/road_image_12.png
inflating: dataset/train/segments/road_image_120.png
inflating: dataset/train/segments/road_image_121.png
inflating: dataset/train/segments/road_image_122.png
inflating: dataset/train/segments/road_image_123.png
inflating: dataset/train/segments/road_image_124.png
inflating: dataset/train/segments/road_image_125.png
inflating: dataset/train/segments/road_image_126.png
inflating: dataset/train/segments/road_image_127.png
inflating: dataset/train/segments/road_image_128.png
inflating: dataset/train/segments/road_image_129.png
inflating: dataset/train/segments/road_image_13.png
inflating: dataset/train/segments/road_image_130.png
inflating: dataset/train/segments/road_image_131.png
inflating: dataset/train/segments/road_image_132.png
inflating: dataset/train/segments/road_image_133.png
inflating: dataset/train/segments/road_image_134.png
inflating: dataset/train/segments/road_image_135.png
inflating: dataset/train/segments/road_image_136.png
inflating: dataset/train/segments/road_image_137.png
inflating: dataset/train/segments/road_image_138.png
inflating: dataset/train/segments/road_image_139.png
inflating: dataset/train/segments/road_image_14.png
inflating: dataset/train/segments/road_image_140.png
inflating: dataset/train/segments/road_image_141.png
inflating: dataset/train/segments/road_image_142.png
inflating: dataset/train/segments/road_image_143.png
inflating: dataset/train/segments/road_image_144.png
inflating: dataset/train/segments/road_image_145.png
inflating: dataset/train/segments/road_image_146.png
inflating: dataset/train/segments/road_image_147.png
inflating: dataset/train/segments/road_image_148.png
inflating: dataset/train/segments/road_image_149.png
```

```
inflating: dataset/train/segments/road_image_15.png
inflating: dataset/train/segments/road_image_150.png
inflating: dataset/train/segments/road_image_151.png
inflating: dataset/train/segments/road_image_152.png
inflating: dataset/train/segments/road_image_153.png
inflating: dataset/train/segments/road_image_154.png
inflating: dataset/train/segments/road_image_155.png
inflating: dataset/train/segments/road_image_156.png
inflating: dataset/train/segments/road_image_157.png
inflating: dataset/train/segments/road_image_158.png
inflating: dataset/train/segments/road_image_159.png
inflating: dataset/train/segments/road_image_16.png
inflating: dataset/train/segments/road_image_17.png
inflating: dataset/train/segments/road_image_18.png
inflating: dataset/train/segments/road_image_19.png
inflating: dataset/train/segments/road_image_2.png
inflating: dataset/train/segments/road_image_20.png
inflating: dataset/train/segments/road_image_21.png
inflating: dataset/train/segments/road_image_22.png
inflating: dataset/train/segments/road_image_23.png
inflating: dataset/train/segments/road_image_24.png
inflating: dataset/train/segments/road_image_25.png
inflating: dataset/train/segments/road_image_26.png
inflating: dataset/train/segments/road_image_27.png
inflating: dataset/train/segments/road_image_28.png
inflating: dataset/train/segments/road_image_29.png
inflating: dataset/train/segments/road_image_3.png
inflating: dataset/train/segments/road_image_30.png
inflating: dataset/train/segments/road_image_31.png
inflating: dataset/train/segments/road_image_32.png
inflating: dataset/train/segments/road_image_33.png
inflating: dataset/train/segments/road_image_34.png
inflating: dataset/train/segments/road_image_35.png
inflating: dataset/train/segments/road_image_36.png
inflating: dataset/train/segments/road_image_37.png
inflating: dataset/train/segments/road_image_38.png
inflating: dataset/train/segments/road_image_39.png
inflating: dataset/train/segments/road_image_4.png
inflating: dataset/train/segments/road_image_40.png
inflating: dataset/train/segments/road_image_41.png
inflating: dataset/train/segments/road_image_42.png
inflating: dataset/train/segments/road_image_43.png
inflating: dataset/train/segments/road_image_44.png
inflating: dataset/train/segments/road_image_45.png
inflating: dataset/train/segments/road_image_46.png
inflating: dataset/train/segments/road_image_47.png
inflating: dataset/train/segments/road_image_48.png
inflating: dataset/train/segments/road_image_49.png
```

```
inflating: dataset/train/segments/road_image_5.png
inflating: dataset/train/segments/road_image_50.png
inflating: dataset/train/segments/road_image_51.png
inflating: dataset/train/segments/road_image_52.png
inflating: dataset/train/segments/road_image_53.png
inflating: dataset/train/segments/road_image_54.png
inflating: dataset/train/segments/road_image_55.png
inflating: dataset/train/segments/road_image_56.png
inflating: dataset/train/segments/road_image_57.png
inflating: dataset/train/segments/road_image_58.png
inflating: dataset/train/segments/road_image_59.png
inflating: dataset/train/segments/road_image_6.png
inflating: dataset/train/segments/road_image_60.png
inflating: dataset/train/segments/road_image_61.png
inflating: dataset/train/segments/road_image_62.png
inflating: dataset/train/segments/road_image_63.png
inflating: dataset/train/segments/road_image_64.png
inflating: dataset/train/segments/road_image_65.png
inflating: dataset/train/segments/road_image_66.png
inflating: dataset/train/segments/road_image_67.png
inflating: dataset/train/segments/road_image_68.png
inflating: dataset/train/segments/road_image_69.png
inflating: dataset/train/segments/road_image_7.png
inflating: dataset/train/segments/road_image_70.png
inflating: dataset/train/segments/road_image_71.png
inflating: dataset/train/segments/road_image_72.png
inflating: dataset/train/segments/road_image_73.png
inflating: dataset/train/segments/road_image_74.png
inflating: dataset/train/segments/road_image_75.png
inflating: dataset/train/segments/road_image_76.png
inflating: dataset/train/segments/road_image_77.png
inflating: dataset/train/segments/road_image_78.png
inflating: dataset/train/segments/road_image_79.png
inflating: dataset/train/segments/road_image_8.png
inflating: dataset/train/segments/road_image_80.png
inflating: dataset/train/segments/road_image_81.png
inflating: dataset/train/segments/road_image_82.png
inflating: dataset/train/segments/road_image_83.png
inflating: dataset/train/segments/road_image_84.png
inflating: dataset/train/segments/road_image_85.png
inflating: dataset/train/segments/road_image_86.png
inflating: dataset/train/segments/road_image_87.png
inflating: dataset/train/segments/road_image_88.png
inflating: dataset/train/segments/road_image_89.png
inflating: dataset/train/segments/road_image_9.png
inflating: dataset/train/segments/road_image_90.png
inflating: dataset/train/segments/road_image_91.png
inflating: dataset/train/segments/road_image_92.png
```

```
inflating: dataset/train/segments/road_image_93.png
inflating: dataset/train/segments/road_image_94.png
inflating: dataset/train/segments/road_image_95.png
inflating: dataset/train/segments/road_image_96.png
inflating: dataset/train/segments/road_image_97.png
inflating: dataset/train/segments/road_image_98.png
inflating: dataset/train/segments/road_image_99.png
 creating: dataset/validation/
 creating: dataset/validation/images/
inflating: dataset/validation/images/road_image_180.png
inflating: dataset/validation/images/road_image_181.png
inflating: dataset/validation/images/road_image_182.png
inflating: dataset/validation/images/road_image_183.png
inflating: dataset/validation/images/road_image_184.png
inflating: dataset/validation/images/road_image_185.png
inflating: dataset/validation/images/road_image_186.png
inflating: dataset/validation/images/road_image_187.png
inflating: dataset/validation/images/road_image_188.png
inflating: dataset/validation/images/road_image_189.png
inflating: dataset/validation/images/road_image_190.png
inflating: dataset/validation/images/road_image_191.png
inflating: dataset/validation/images/road_image_192.png
inflating: dataset/validation/images/road_image_193.png
inflating: dataset/validation/images/road_image_194.png
inflating: dataset/validation/images/road_image_195.png
inflating: dataset/validation/images/road_image_196.png
inflating: dataset/validation/images/road_image_197.png
inflating: dataset/validation/images/road_image_198.png
inflating: dataset/validation/images/road_image_199.png
 creating: dataset/validation/lane/
inflating: dataset/validation/lane/road_image_180.png
inflating: dataset/validation/lane/road_image_181.png
inflating: dataset/validation/lane/road_image_182.png
inflating: dataset/validation/lane/road_image_183.png
inflating: dataset/validation/lane/road_image_184.png
inflating: dataset/validation/lane/road_image_185.png
inflating: dataset/validation/lane/road_image_186.png
inflating: dataset/validation/lane/road_image_187.png
inflating: dataset/validation/lane/road_image_188.png
inflating: dataset/validation/lane/road_image_189.png
inflating: dataset/validation/lane/road_image_190.png
inflating: dataset/validation/lane/road_image_191.png
inflating: dataset/validation/lane/road_image_192.png
inflating: dataset/validation/lane/road_image_193.png
inflating: dataset/validation/lane/road_image_194.png
inflating: dataset/validation/lane/road_image_195.png
inflating: dataset/validation/lane/road_image_196.png
inflating: dataset/validation/lane/road_image_197.png
```

```
  inflating: dataset/validation/lane/road_image_198.png
  inflating: dataset/validation/lane/road_image_199.png
   creating: dataset/validation/segments/
  inflating: dataset/validation/segments/road_image_180.png
  inflating: dataset/validation/segments/road_image_181.png
  inflating: dataset/validation/segments/road_image_182.png
  inflating: dataset/validation/segments/road_image_183.png
  inflating: dataset/validation/segments/road_image_184.png
  inflating: dataset/validation/segments/road_image_185.png
  inflating: dataset/validation/segments/road_image_186.png
  inflating: dataset/validation/segments/road_image_187.png
  inflating: dataset/validation/segments/road_image_188.png
  inflating: dataset/validation/segments/road_image_189.png
  inflating: dataset/validation/segments/road_image_190.png
  inflating: dataset/validation/segments/road_image_191.png
  inflating: dataset/validation/segments/road_image_192.png
  inflating: dataset/validation/segments/road_image_193.png
  inflating: dataset/validation/segments/road_image_194.png
  inflating: dataset/validation/segments/road_image_195.png
  inflating: dataset/validation/segments/road_image_196.png
  inflating: dataset/validation/segments/road_image_197.png
  inflating: dataset/validation/segments/road_image_198.png
  inflating: dataset/validation/segments/road_image_199.png
```

0.5 Import the all the required libraries

[3]:
```
import torch
import cv2
import torch.utils.data
import torchvision.transforms as transforms
import numpy as np
import os
import random
import math
from matplotlib import pyplot as plt
import torch.nn as nn
```

0.6 Image transformation functions

- By paper author

[4]:
```
def augment_hsv(img, hgain=0.015, sgain=0.7, vgain=0.4):
    """change color hue, saturation, value"""
    r = np.random.uniform(-1, 1, 3) * [hgain, sgain, vgain] + 1  # random gains
    hue, sat, val = cv2.split(cv2.cvtColor(img, cv2.COLOR_BGR2HSV))
    dtype = img.dtype  # uint8
```

```
    x = np.arange(0, 256, dtype=np.int16)
    lut_hue = ((x * r[0]) % 180).astype(dtype)
    lut_sat = np.clip(x * r[1], 0, 255).astype(dtype)
    lut_val = np.clip(x * r[2], 0, 255).astype(dtype)

    img_hsv = cv2.merge((cv2.LUT(hue, lut_hue), cv2.LUT(sat, lut_sat), cv2.
↪LUT(val, lut_val))).astype(dtype)
    cv2.cvtColor(img_hsv, cv2.COLOR_HSV2BGR, dst=img)  # no return needed
```

[5]:
```
def random_perspective(combination,  degrees=10, translate=.1, scale=.1,␣
↪shear=10, perspective=0.0, border=(0, 0)):
    """combination of img transform"""
    # torchvision.transforms.RandomAffine(degrees=(-10, 10), translate=(.1, .
↪1), scale=(.9, 1.1), shear=(-10, 10))
    # targets = [cls, xyxy]
    img, gray, line = combination
    height = img.shape[0] + border[0] * 2  # shape(h,w,c)
    width = img.shape[1] + border[1] * 2

    # Center
    C = np.eye(3)
    C[0, 2] = -img.shape[1] / 2  # x translation (pixels)
    C[1, 2] = -img.shape[0] / 2  # y translation (pixels)

    # Perspective
    P = np.eye(3)
    P[2, 0] = random.uniform(-perspective, perspective)  # x perspective (about␣
↪y)
    P[2, 1] = random.uniform(-perspective, perspective)  # y perspective (about␣
↪x)

    # Rotation and Scale
    R = np.eye(3)
    a = random.uniform(-degrees, degrees)
    # a += random.choice([-180, -90, 0, 90])  # add 90deg rotations to small␣
↪rotations
    s = random.uniform(1 - scale, 1 + scale)
    # s = 2 ** random.uniform(-scale, scale)
    R[:2] = cv2.getRotationMatrix2D(angle=a, center=(0, 0), scale=s)

    # Shear
    S = np.eye(3)
    S[0, 1] = math.tan(random.uniform(-shear, shear) * math.pi / 180)  # x␣
↪shear (deg)
    S[1, 0] = math.tan(random.uniform(-shear, shear) * math.pi / 180)  # y␣
↪shear (deg)
```

```
    # Translation
    T = np.eye(3)
    T[0, 2] = random.uniform(0.5 - translate, 0.5 + translate) * width  # x␣
↪translation (pixels)
    T[1, 2] = random.uniform(0.5 - translate, 0.5 + translate) * height  # y␣
↪translation (pixels)

    # Combined rotation matrix
    M = T @ S @ R @ P @ C  # order of operations (right to left) is IMPORTANT
    if (border[0] != 0) or (border[1] != 0) or (M != np.eye(3)).any():  # image␣
↪changed
        if perspective:
            img = cv2.warpPerspective(img, M, dsize=(width, height),␣
↪borderValue=(114, 114, 114))
            gray = cv2.warpPerspective(gray, M, dsize=(width, height),␣
↪borderValue=0)
            line = cv2.warpPerspective(line, M, dsize=(width, height),␣
↪borderValue=0)
        else:  # affine
            img = cv2.warpAffine(img, M[:2], dsize=(width, height),␣
↪borderValue=(114, 114, 114))
            gray = cv2.warpAffine(gray, M[:2], dsize=(width, height),␣
↪borderValue=0)
            line = cv2.warpAffine(line, M[:2], dsize=(width, height),␣
↪borderValue=0)



    combination = (img, gray, line)
    return combination
```

[6]:
```
!mkdir custom-dataset
!mkdir custom-dataset/test
!mkdir custom-dataset/train
!mkdir custom-dataset/validation

!mkdir -p custom-dataset/train/images
!mkdir -p custom-dataset/train/lane
!mkdir -p custom-dataset/train/segments
!mkdir -p custom-dataset/test/images
!mkdir -p custom-dataset/test/lane
!mkdir -p custom-dataset/test/segments
!mkdir -p custom-dataset/validation/images
!mkdir -p custom-dataset/validation/lane
!mkdir -p custom-dataset/validation/segments
```

```
# Copy only two images from each category in train
!cp dataset/train/images/road_image_0.png custom-dataset/train/images/
!cp dataset/train/lane/road_image_0.png custom-dataset/train/lane/
!cp dataset/train/segments/road_image_0.png custom-dataset/train/segments/

# Copy only two images from each category in train
!cp dataset/train/images/road_image_1.png custom-dataset/train/images/
!cp dataset/train/lane/road_image_1.png custom-dataset/train/lane/
!cp dataset/train/segments/road_image_1.png custom-dataset/train/segments/

# Copy only two images from each category in test
!cp dataset/test/images/* custom-dataset/test/images/
!cp dataset/test/lane/* custom-dataset/test/lane/
!cp dataset/test/segments/* custom-dataset/test/segments/


# Copy only two images from each category in validation
!cp dataset/validation/images/* custom-dataset/validation/images/
!cp dataset/validation/lane/* custom-dataset/validation/lane/
!cp dataset/validation/segments/* custom-dataset/validation/segments/
```

0.7 Custom Dataset Class

- This custom dataset class is based on the dataset class written by the author but with slight modifications like path. we have adjusted the path according to the google colab.

[7]:
```
class MyDataset(torch.utils.data.Dataset):
    '''
    Class to load the dataset
    '''
    def __init__(self, transform=None, valid=False, test=False):
        '''
        :param imList: image list (Note that these lists have been processed␣
 ↪and pickled using the loadData.py)
        :param labelList: label list (Note that these lists have been processed␣
 ↪and pickled using the loadData.py)
        :param transform: Type of transformation. SEe Transforms.py for␣
 ↪supported transformations
        '''

        self.transform = transform
        self.Tensor = transforms.ToTensor()
        self.valid=valid
        if valid:
            self.root='custom-dataset/validation/images'
            self.names=os.listdir(self.root)
```

```
        elif test:
            self.root='custom-dataset/test/images'
            self.names=os.listdir(self.root)
        else:
            self.root='custom-dataset/train/images/'
            self.names=os.listdir(self.root)

    def __len__(self):
        return len(self.names)

    def __getitem__(self, idx):
        '''

        :param idx: Index of the image file
        :return: returns the image and corresponding label file.
        '''
        W_=640
        H_=360
        image_name=os.path.join(self.root,self.names[idx])

        image = cv2.imread(image_name)
        original_image = cv2.imread(image_name)
        label1 = cv2.imread(image_name.replace("images","segments").
↪replace("jpg","png"), 0)
        label2 = cv2.imread(image_name.replace("images","lane").
↪replace("jpg","png"), 0)
        if not self.valid:
            if random.random()<0.5:
                combination = (image, label1, label2)
                (image, label1, label2)= random_perspective(
                    combination=combination,
                    degrees=10,
                    translate=0.1,
                    scale=0.25,
                    shear=0.0
                )
            if random.random()<0.5:
                augment_hsv(image)
            if random.random() < 0.5:
                image = np.fliplr(image)
                label1 = np.fliplr(label1)
                label2 = np.fliplr(label2)

        label1 = cv2.resize(label1, (W_, H_))
        label2 = cv2.resize(label2, (W_, H_))
        image = cv2.resize(image, (W_, H_))
```

```
        _,seg_b1 = cv2.threshold(label1,1,255,cv2.THRESH_BINARY_INV)
        _,seg_b2 = cv2.threshold(label2,1,255,cv2.THRESH_BINARY_INV)
        _,seg1 = cv2.threshold(label1,1,255,cv2.THRESH_BINARY)
        _,seg2 = cv2.threshold(label2,1,255,cv2.THRESH_BINARY)

        seg1 = self.Tensor(seg1)
        seg2 = self.Tensor(seg2)
        seg_b1 = self.Tensor(seg_b1)
        seg_b2 = self.Tensor(seg_b2)
        seg_da = torch.stack((seg_b1[0], seg1[0]),0)
        seg_ll = torch.stack((seg_b2[0], seg2[0]),0)
        image = image[:, :, ::-1].transpose(2, 0, 1)
        image = np.ascontiguousarray(image)

        return original_image, image_name,torch.
↪from_numpy(image),(seg_da,seg_ll)
```

0.8 Intialize a dataloader

- Intialize a dataloader with batch size 8
- Intialize train, test, validation datasets.

[71]:
```
from torch.utils.data import DataLoader

train_dataloader = DataLoader(MyDataset(), batch_size = 2, shuffle = True)
test_dataloader = DataLoader(MyDataset(test=True), batch_size = 8, shuffle =␣
↪True)
val_dataloader = DataLoader(MyDataset(valid=True), batch_size = 8, shuffle =␣
↪True)
```

0.9 Take only two samples

[72]:
```
_, _, input, target = next(iter(train_dataloader))
```

0.10 Copy necessary files from repository

[10]:
```
# Copy pretrained model from repository to root
!cp road-detection/TwinLiteNet/pretrained/best.pth ./

# Copy pytorch Neural Net from repo to root
!cp road-detection/TwinLiteNet/model/TwinLite.py ./

# Copy Loss function pytorch code from repo to root
!cp road-detection/TwinLiteNet/loss.py ./

# Copy all reqired constants from repo to root
```

```
!cp road-detection/TwinLiteNet/const.py ./

# Copy all val.py from repo to root
!cp road-detection/TwinLiteNet/val.py ./
```

0.11 Mini Version of Original Network

[73]:
```
import torch.nn as nn
import torch.nn.functional as F

class MiniNet(nn.Module):
    def __init__(self):
        super(MiniNet, self).__init__()

        # Input Convolution
        self.conv1 = nn.Conv2d(3, 16, kernel_size=3, stride=2, padding=1)
        self.bn1 = nn.BatchNorm2d(16)
        self.relu1 = nn.ReLU()

        # Downsampling
        self.pool = nn.MaxPool2d(kernel_size=3, stride=2, padding=1)

        # Intermediate Convolutions
        self.conv2 = nn.Conv2d(16, 32, kernel_size=3, stride=1, padding=1)
        self.bn2 = nn.BatchNorm2d(32)
        self.relu2 = nn.ReLU()

        self.conv3 = nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1)
        self.bn3 = nn.BatchNorm2d(64)
        self.relu3 = nn.ReLU()

        # Upsampling
        self.upconv1 = nn.ConvTranspose2d(64, 32, kernel_size=4, stride=2,␣
 ↪padding=1)
        self.bn4 = nn.BatchNorm2d(32)
        self.relu4 = nn.ReLU()

        self.upconv2 = nn.ConvTranspose2d(32, 16, kernel_size=4, stride=2,␣
 ↪padding=1)
        self.bn5 = nn.BatchNorm2d(16)
        self.relu5 = nn.ReLU()

        # Classifiers
        self.classifier1 = nn.Conv2d(16, 2, kernel_size=1)
        self.classifier2 = nn.Conv2d(16, 2, kernel_size=1)
```

```
    def forward(self, x):
        # Input Convolution
        x = self.relu1(self.bn1(self.conv1(x)))

        # Downsampling
        x = self.pool(x)

        # Intermediate Convolutions
        x = self.relu2(self.bn2(self.conv2(x)))
        x = self.relu3(self.bn3(self.conv3(x)))

        # Upsampling
        x = self.relu4(self.bn4(self.upconv1(x)))
        x = self.relu5(self.bn5(self.upconv2(x)))

        # Classifiers
        classifier1 = self.classifier1(x)
        classifier2 = self.classifier2(x)

        return classifier1, classifier2

# Instantiate the model
model = MiniNet().to("cuda")
```

0.12 Defining Optimizer and loss for training

[74]:
```
from loss import TotalLoss
```

[75]:
```
optimizer = torch.optim.Adam(model.parameters(), 5e-4, (0.9, 0.999), eps=1e-08,␣
 ↪weight_decay=5e-4)
criterion = TotalLoss()
```

0.13 Defining Custom metrics for evaluation

[76]:
```
from tqdm import tqdm

class SegmentationMetric(object):
    '''
    imgLabel [batch_size, height(144), width(256)]
    confusionMatrix [[0(TN),1(FP)],
                     [2(FN),3(TP)]]
    '''
    def __init__(self, numClass):
        self.numClass = numClass
        self.confusionMatrix = np.zeros((self.numClass,)*2)
```

```
    def pixelAccuracy(self):
        # return all class overall pixel accuracy
        # acc = (TP + TN) / (TP + TN + FP + TN)
        acc = np.diag(self.confusionMatrix).sum() /  self.confusionMatrix.sum()
        return acc


    def classPixelAccuracy(self):
        # return each category pixel accuracy(A more accurate way to call it␣
↪precision)
        # acc = (TP) / TP + FP
        classAcc = np.diag(self.confusionMatrix) / (self.confusionMatrix.
↪sum(axis=0) + 1e-12)
        return classAcc

    def meanPixelAccuracy(self):
        classAcc = self.classPixelAccuracy()
        meanAcc = np.nanmean(classAcc)
        return meanAcc

    def meanIntersectionOverUnion(self):
        # Intersection = TP Union = TP + FP + FN
        # IoU = TP / (TP + FP + FN)
        intersection = np.diag(self.confusionMatrix)
        union = np.sum(self.confusionMatrix, axis=1) + np.sum(self.
↪confusionMatrix, axis=0) - np.diag(self.confusionMatrix)
        IoU = intersection / union
        IoU[np.isnan(IoU)] = 0
        mIoU = np.nanmean(IoU)
        return mIoU

    def IntersectionOverUnion(self):
        intersection = np.diag(self.confusionMatrix)
        union = np.sum(self.confusionMatrix, axis=1) + np.sum(self.
↪confusionMatrix, axis=0) - np.diag(self.confusionMatrix)
        IoU = intersection / union
        IoU[np.isnan(IoU)] = 0
        return IoU[1]

    def genConfusionMatrix(self, imgPredict, imgLabel):
        # remove classes from unlabeled pixels in gt image and predict
        # print(imgLabel.shape)
        mask = (imgLabel >= 0) & (imgLabel < self.numClass)
        label = self.numClass * imgLabel[mask] + imgPredict[mask]
        count = np.bincount(label, minlength=self.numClass**2)
        confusionMatrix = count.reshape(self.numClass, self.numClass)
        return confusionMatrix
```

```
    def Frequency_Weighted_Intersection_over_Union(self):
        # FWIOU =     [(TP+FN)/(TP+FP+TN+FN)] *[TP / (TP + FP + FN)]
        freq = np.sum(self.confusionMatrix, axis=1) / np.sum(self.
 ↪confusionMatrix)
        iu = np.diag(self.confusionMatrix) / (
                np.sum(self.confusionMatrix, axis=1) + np.sum(self.
 ↪confusionMatrix, axis=0) -
                np.diag(self.confusionMatrix))
        FWIoU = (freq[freq > 0] * iu[freq > 0]).sum()
        return FWIoU


    def addBatch(self, imgPredict, imgLabel):
        assert imgPredict.shape == imgLabel.shape
        self.confusionMatrix += self.genConfusionMatrix(imgPredict, imgLabel)

    def reset(self):
        self.confusionMatrix = np.zeros((self.numClass, self.numClass))

class AverageMeter(object):
    """Computes and stores the average and current value"""
    def __init__(self):
        self.reset()

    def reset(self):
        self.val = 0
        self.avg = 0
        self.sum = 0
        self.count = 0

    def update(self, val, n=1):
        self.val = val
        self.sum += val * n
        self.count += n
        self.avg = self.sum / self.count if self.count != 0 else 0

@torch.no_grad()
def val(val_loader, model):

    model.eval()


    DA=SegmentationMetric(2)
    LL=SegmentationMetric(2)

    da_acc_seg = AverageMeter()
```

```
    da_IoU_seg = AverageMeter()
    da_mIoU_seg = AverageMeter()

    ll_acc_seg = AverageMeter()
    ll_IoU_seg = AverageMeter()
    ll_mIoU_seg = AverageMeter()
    total_batches = len(val_loader)

    total_batches = len(val_loader)
    pbar = enumerate(val_loader)
    pbar = tqdm(pbar, total=total_batches)
    for i, (_, _,input, target) in pbar:
        input = input.cuda().float() / 255.0
            # target = target.cuda()

        input_var = input
        target_var = target

        # run the mdoel
        with torch.no_grad():
            output = model(input_var)

        out_da,out_ll=output
        target_da,target_ll=target

        _,da_predict=torch.max(out_da, 1)
        _,da_gt=torch.max(target_da, 1)

        _,ll_predict=torch.max(out_ll, 1)
        _,ll_gt=torch.max(target_ll, 1)
        DA.reset()
        DA.addBatch(da_predict.cpu(), da_gt.cpu())


        da_acc = DA.pixelAccuracy()
        da_IoU = DA.IntersectionOverUnion()
        da_mIoU = DA.meanIntersectionOverUnion()

        da_acc_seg.update(da_acc,input.size(0))
        da_IoU_seg.update(da_IoU,input.size(0))
        da_mIoU_seg.update(da_mIoU,input.size(0))


        LL.reset()
        LL.addBatch(ll_predict.cpu(), ll_gt.cpu())
```

```
        ll_acc = LL.pixelAccuracy()
        ll_IoU = LL.IntersectionOverUnion()
        ll_mIoU = LL.meanIntersectionOverUnion()

        ll_acc_seg.update(ll_acc,input.size(0))
        ll_IoU_seg.update(ll_IoU,input.size(0))
        ll_mIoU_seg.update(ll_mIoU,input.size(0))

    da_segment_result = (da_acc_seg.avg,da_IoU_seg.avg,da_mIoU_seg.avg)
    ll_segment_result = (ll_acc_seg.avg,ll_IoU_seg.avg,ll_mIoU_seg.avg)
    return da_segment_result,ll_segment_result
```

0.14 Training The model

[77]:
```
val_loss = []
train_loss = []

for i in list(range(1000)):
    model.train()
    model.to("cuda")
    output = model(input.cuda().float() / 255.0)

    # target=target.cuda()
    # print(target[0].size())
    optimizer.zero_grad()
    focal_loss, tversky_loss, loss = criterion(output,target)
    optimizer.zero_grad()
    train_loss.append(loss.item())
    # print(output.size())
    loss.backward()
    optimizer.step()
    if i % 50 == 0:
        print("loss: {loss:.5f}".format(loss = loss.item()))
        model.eval()
        example = torch.rand(1, 3, 360, 640).cuda()
        model = torch.jit.trace(model, example)
        print("Accuracy:")
        da_segment_results,ll_segment_results = val(train_dataloader, model)

        msg =  'Driving area Segment: Acc({da_seg_acc:.3f})\n' \
                            'Lane line Segment: Acc({ll_seg_acc:.3f})'.format(
                                da_seg_acc=da_segment_results[0],
                                ll_seg_acc=ll_segment_results[0])
        print(msg)
        print()
        print("Validation Evaluation:")
        da_segment_results,ll_segment_results = val(val_dataloader, model)
```

```
      msg =  'Driving area Segment: Acc({da_seg_acc:.3f})    IOU ({da_seg_iou:
↪.3f})    mIOU({da_seg_miou:.3f})\n' \
                           'Lane line Segment: Acc({ll_seg_acc:.3f})    IOU␣
↪({ll_seg_iou:.3f})  mIOU({ll_seg_miou:.3f})'.format(
                                ␣
↪da_seg_acc=da_segment_results[0],da_seg_iou=da_segment_results[1],da_seg_miou=da_segment_re
                                ␣
↪ll_seg_acc=ll_segment_results[0],ll_seg_iou=ll_segment_results[1],ll_seg_miou=ll_segment_re
      print(msg)
     ␣
↪print("-------------------------------------------------------------------")
```

```
loss: 1.38184
Accuracy:
```

```
100%|      | 1/1 [00:00<00:00,  1.01it/s]
```

```
Driving area Segment: Acc(0.102)
Lane line Segment: Acc(0.005)

Validation Evaluation:
```

```
100%|      | 3/3 [00:04<00:00,  1.52s/it]
```

```
Driving area Segment: Acc(0.202)    IOU (0.202)    mIOU(0.101)
Lane line Segment: Acc(0.019)    IOU (0.019)  mIOU(0.009)
-------------------------------------------------------------------

/usr/local/lib/python3.10/dist-packages/torch/jit/_trace.py:787: UserWarning:
The input to trace is already a ScriptModule, tracing it is a no-op. Returning
the object as is.
  warnings.warn(
```

```
loss: 0.94735
Accuracy:
```

```
100%|      | 1/1 [00:00<00:00,  8.81it/s]
```

```
Driving area Segment: Acc(0.856)
Lane line Segment: Acc(0.993)

Validation Evaluation:
```

```
100%|      | 3/3 [00:01<00:00,  2.09it/s]
```

```
Driving area Segment: Acc(0.797)    IOU (0.000)    mIOU(0.399)
Lane line Segment: Acc(0.981)    IOU (0.000)  mIOU(0.491)
-------------------------------------------------------------------
loss: 0.81066
Accuracy:
```

```
100%|      | 1/1 [00:00<00:00,  8.84it/s]
```

```
Driving area Segment: Acc(0.839)
Lane line Segment: Acc(0.992)

Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.13it/s]

Driving area Segment: Acc(0.798)    IOU (0.000)    mIOU(0.399)
Lane line Segment: Acc(0.981)    IOU (0.000)  mIOU(0.491)
---------------------------------------------------------------
loss: 0.67491
Accuracy:

100%|      | 1/1 [00:00<00:00,  8.64it/s]

Driving area Segment: Acc(0.874)
Lane line Segment: Acc(0.993)

Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.07it/s]

Driving area Segment: Acc(0.785)    IOU (0.027)    mIOU(0.405)
Lane line Segment: Acc(0.981)    IOU (0.000)  mIOU(0.491)
---------------------------------------------------------------
loss: 0.29812
Accuracy:

100%|      | 1/1 [00:00<00:00,  5.03it/s]

Driving area Segment: Acc(0.922)
Lane line Segment: Acc(0.995)

Validation Evaluation:

100%|      | 3/3 [00:02<00:00,  1.41it/s]

Driving area Segment: Acc(0.808)    IOU (0.245)    mIOU(0.519)
Lane line Segment: Acc(0.971)    IOU (0.021)  mIOU(0.496)
---------------------------------------------------------------
loss: 0.17488
Accuracy:

100%|      | 1/1 [00:00<00:00,  8.60it/s]

Driving area Segment: Acc(0.950)
Lane line Segment: Acc(0.994)

Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.13it/s]

Driving area Segment: Acc(0.835)    IOU (0.332)    mIOU(0.577)
Lane line Segment: Acc(0.969)    IOU (0.029)  mIOU(0.499)
---------------------------------------------------------------
```

```
loss: 0.12239
Accuracy:
100%|      | 1/1 [00:00<00:00,  8.60it/s]
Driving area Segment: Acc(0.958)
Lane line Segment: Acc(0.995)

Validation Evaluation:
100%|      | 3/3 [00:01<00:00,  2.07it/s]
Driving area Segment: Acc(0.853)    IOU (0.366)    mIOU(0.602)
Lane line Segment: Acc(0.965)    IOU (0.037)  mIOU(0.501)
------------------------------------------------------------
loss: 0.09719
Accuracy:
100%|      | 1/1 [00:00<00:00,  8.13it/s]
Driving area Segment: Acc(0.919)
Lane line Segment: Acc(0.988)

Validation Evaluation:
100%|      | 3/3 [00:01<00:00,  2.19it/s]
Driving area Segment: Acc(0.846)    IOU (0.301)    mIOU(0.568)
Lane line Segment: Acc(0.965)    IOU (0.035)  mIOU(0.500)
------------------------------------------------------------
loss: 0.08525
Accuracy:
100%|      | 1/1 [00:00<00:00,  5.92it/s]
Driving area Segment: Acc(0.970)
Lane line Segment: Acc(0.995)

Validation Evaluation:
100%|      | 3/3 [00:02<00:00,  1.39it/s]
Driving area Segment: Acc(0.851)    IOU (0.312)    mIOU(0.576)
Lane line Segment: Acc(0.964)    IOU (0.044)  mIOU(0.504)
------------------------------------------------------------
loss: 0.06760
Accuracy:
100%|      | 1/1 [00:00<00:00,  8.43it/s]
Driving area Segment: Acc(0.923)
Lane line Segment: Acc(0.997)

Validation Evaluation:
```

```
100%|      | 3/3 [00:01<00:00,  2.11it/s]

Driving area Segment: Acc(0.849)    IOU (0.298)    mIOU(0.568)
Lane line Segment: Acc(0.963)    IOU (0.034)  mIOU(0.499)
---------------------------------------------------------------
loss: 0.04619
Accuracy:

100%|      | 1/1 [00:00<00:00,  8.42it/s]

Driving area Segment: Acc(0.955)
Lane line Segment: Acc(0.990)


Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.11it/s]

Driving area Segment: Acc(0.838)    IOU (0.222)    mIOU(0.526)
Lane line Segment: Acc(0.966)    IOU (0.054)  mIOU(0.510)
---------------------------------------------------------------
loss: 0.04413
Accuracy:

100%|      | 1/1 [00:00<00:00,  8.36it/s]

Driving area Segment: Acc(0.975)
Lane line Segment: Acc(0.997)


Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.15it/s]

Driving area Segment: Acc(0.837)    IOU (0.218)    mIOU(0.524)
Lane line Segment: Acc(0.967)    IOU (0.040)  mIOU(0.503)
---------------------------------------------------------------
loss: 0.03592
Accuracy:

100%|      | 1/1 [00:00<00:00,  7.39it/s]

Driving area Segment: Acc(0.978)
Lane line Segment: Acc(0.997)


Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  1.83it/s]

Driving area Segment: Acc(0.843)    IOU (0.250)    mIOU(0.542)
Lane line Segment: Acc(0.966)    IOU (0.051)  mIOU(0.509)
---------------------------------------------------------------
loss: 0.03287
Accuracy:

100%|      | 1/1 [00:00<00:00,  5.25it/s]
```

```
Driving area Segment: Acc(0.976)
Lane line Segment: Acc(0.997)

Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  1.73it/s]

Driving area Segment: Acc(0.845)    IOU (0.239)    mIOU(0.537)
Lane line Segment: Acc(0.966)    IOU (0.060)  mIOU(0.513)
---------------------------------------------------------------
loss: 0.03050
Accuracy:

100%|      | 1/1 [00:00<00:00,  8.60it/s]

Driving area Segment: Acc(0.962)
Lane line Segment: Acc(0.996)

Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.23it/s]

Driving area Segment: Acc(0.846)    IOU (0.266)    mIOU(0.551)
Lane line Segment: Acc(0.966)    IOU (0.060)  mIOU(0.513)
---------------------------------------------------------------
loss: 0.02859
Accuracy:

100%|      | 1/1 [00:00<00:00,  8.35it/s]

Driving area Segment: Acc(0.976)
Lane line Segment: Acc(0.997)

Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.17it/s]

Driving area Segment: Acc(0.846)    IOU (0.265)    mIOU(0.551)
Lane line Segment: Acc(0.967)    IOU (0.047)  mIOU(0.507)
---------------------------------------------------------------
loss: 0.03143
Accuracy:

100%|      | 1/1 [00:00<00:00,  7.43it/s]

Driving area Segment: Acc(0.949)
Lane line Segment: Acc(0.997)

Validation Evaluation:

100%|      | 3/3 [00:01<00:00,  2.29it/s]

Driving area Segment: Acc(0.860)    IOU (0.343)    mIOU(0.596)
Lane line Segment: Acc(0.965)    IOU (0.064)  mIOU(0.514)
---------------------------------------------------------------
```

```
loss: 0.02626
Accuracy:
100%|      | 1/1 [00:00<00:00,  5.28it/s]
Driving area Segment: Acc(0.973)
Lane line Segment: Acc(0.997)

Validation Evaluation:
100%|      | 3/3 [00:02<00:00,  1.27it/s]
Driving area Segment: Acc(0.847)    IOU (0.267)    mIOU(0.552)
Lane line Segment: Acc(0.969)    IOU (0.051)  mIOU(0.510)
------------------------------------------------------------
loss: 0.02511
Accuracy:
100%|      | 1/1 [00:00<00:00,  8.58it/s]
Driving area Segment: Acc(0.978)
Lane line Segment: Acc(0.998)

Validation Evaluation:
100%|      | 3/3 [00:01<00:00,  1.59it/s]
Driving area Segment: Acc(0.849)    IOU (0.284)    mIOU(0.561)
Lane line Segment: Acc(0.968)    IOU (0.072)  mIOU(0.520)
------------------------------------------------------------
loss: 0.02446
Accuracy:
100%|      | 1/1 [00:00<00:00,  7.59it/s]
Driving area Segment: Acc(0.982)
Lane line Segment: Acc(0.997)

Validation Evaluation:
100%|      | 3/3 [00:01<00:00,  2.09it/s]
Driving area Segment: Acc(0.851)    IOU (0.298)    mIOU(0.570)
Lane line Segment: Acc(0.968)    IOU (0.059)  mIOU(0.514)
------------------------------------------------------------
```

0.15 Evaluating the model on the test set

[79]:
```
print("Test Evaluation:")
da_segment_results,ll_segment_results = val(test_dataloader, model)

msg =  '\nDriving area Segment: Acc({da_seg_acc:.3f})    IOU ({da_seg_iou:.3f})␣
 ↪   mIOU({da_seg_miou:.3f})\n' \
```

```
                    'Lane line Segment: Acc({ll_seg_acc:.3f})    IOU␣
 ↪({ll_seg_iou:.3f})  mIOU({ll_seg_miou:.3f})'.format(
                         ␣
 ↪da_seg_acc=da_segment_results[0],da_seg_iou=da_segment_results[1],da_seg_miou=da_segment_re
                         ␣
 ↪ll_seg_acc=ll_segment_results[0],ll_seg_iou=ll_segment_results[1],ll_seg_miou=ll_segment_re
print(msg)
print("-----------------------------------------------------------------")
```

```
Test Evaluation:
100%|      | 3/3 [00:02<00:00,  1.25it/s]

Driving area Segment: Acc(0.867)    IOU (0.176)    mIOU(0.520)
Lane line Segment: Acc(0.978)    IOU (0.023)  mIOU(0.500)
-----------------------------------------------------------------
```