Update_2

November 28, 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

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

```
Cloning into 'road-detection'…
remote: Enumerating objects: 441, done.
remote: Counting objects: 100% (182/182), done.
remote: Compressing objects: 100% (162/162), done.
remote: Total 441 (delta 62), reused 49 (delta 17), pack-reused 259
Receiving objects: 100% (441/441), 204.71 MiB | 16.66 MiB/s, done.
Resolving deltas: 100% (155/155), 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"

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

```
Requirement already satisfied: certifi==2023.7.22 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 1)) (2023.7.22)
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 6.4 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
5.0 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
5.6 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 14.1 MB/s eta 0:00:00
Requirement already satisfied: idna==3.4 in
/usr/local/lib/python3.10/dist-packages (from -r road-
detection/TwinLiteNet/requirements.txt (line 11)) (3.4)
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 36.8 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 47.4 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
46.2 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
9.5 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
105.3 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
13.4 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
78.2 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
50.4 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+cu118)
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 936.7 kB/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+cu118)
```

```
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=f339be183854359130f20a0e497726b5480925d6a40d5e616bd30effeb5fe1b0
  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, fsspec, fonttools, dnspython, colorama, 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: 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 4.44.3
    Uninstalling fonttools-4.44.3:
      Successfully uninstalled fonttools-4.44.3
  Attempting uninstall: scipy
    Found existing installation: scipy 1.11.3
    Uninstalling scipy-1.11.3:
      Successfully uninstalled scipy-1.11.3
  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 colorama-0.4.6 dnspython-2.4.2
elephant-0.12.0 fonttools-4.44.0 fsspec-2023.10.0 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

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

0.4 Unzip the file

[ ]:
```
!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

[31]:
```
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

[32]:
```
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
```

[33]:
```
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
```

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.

[34]:
```
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='dataset/validation/images'
            self.names=os.listdir(self.root)
        elif test:
            self.root='dataset/test/images'
            self.names=os.listdir(self.root)
        else:
            self.root='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.

[35]:
```
from torch.utils.data import DataLoader

train_dataloader = DataLoader(MyDataset(), batch_size = 8, 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 Display images

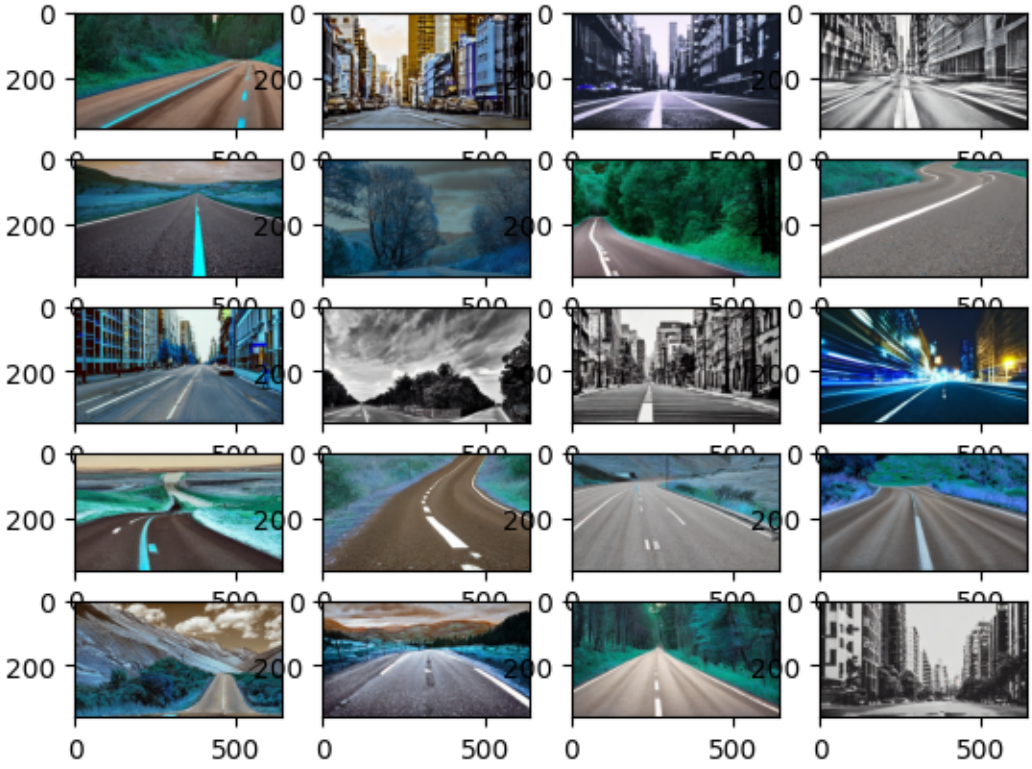
- Show first sample of each mini-batch with size 8

[36]:
```
# Printing the first sample of the each minibatch of size 8

plt.figure(figsize = (100, 100))

f, axarr = plt.subplots(5, 4)
i = 0
j = 0

for batch in train_dataloader:
  original_image, image_name, input, target = batch
  print(image_name[0])
  axarr[i, j].imshow(original_image[0])
  j += 1
  if j%4 == 0:
    i += 1
    j = 0

plt.show()
```

```
dataset/train/images/road_image_99.png
dataset/train/images/road_image_108.png
dataset/train/images/road_image_105.png
dataset/train/images/road_image_143.png
dataset/train/images/road_image_6.png
dataset/train/images/road_image_22.png
dataset/train/images/road_image_34.png
dataset/train/images/road_image_39.png
```

```
dataset/train/images/road_image_149.png
dataset/train/images/road_image_154.png
dataset/train/images/road_image_155.png
dataset/train/images/road_image_138.png
dataset/train/images/road_image_7.png
dataset/train/images/road_image_41.png
dataset/train/images/road_image_92.png
dataset/train/images/road_image_93.png
dataset/train/images/road_image_9.png
dataset/train/images/road_image_45.png
dataset/train/images/road_image_16.png
dataset/train/images/road_image_107.png
```

```
<Figure size 10000x10000 with 0 Axes>
```



0.10 Copy the required files from the repository to Root

[ ]:
```
# 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 Load the pretrained model

[67]:
```
import TwinLite as net


model = net.TwinLiteNet()
model = torch.nn.DataParallel(model)
model = model.cuda()
model.load_state_dict(torch.load('best.pth'))
```

[67]:
```
<All keys matched successfully>
```

0.12 Intialize loss and optimizer.

- This is based on the original code from paper author

[68]:
```
from tqdm import tqdm
from loss import TotalLoss

lr = 5e-4
optimizer = torch.optim.Adam(model.parameters(), lr, (0.9, 0.999), eps=1e-08,␣
 ↪weight_decay=5e-4)

criteria = TotalLoss()
```

[69]:
```
args = dict()

args["lr"] = lr
args["max_epochs"] = 8
args["onGPU"] = True
```

[70]:
```
args
```

[70]:
```
{'lr': 0.0005, 'max_epochs': 8, 'onGPU': True}
```

0.13 Intialize Polynomial Learning Rate Scheduler

- By Paper Author

[71]:
```
def poly_lr_scheduler(args, optimizer, epoch, power=2):
    lr = round(args["lr"] * (1 - epoch / args["max_epochs"]) ** power, 8)
    for param_group in optimizer.param_groups:
        param_group['lr'] = lr

    return lr
```

0.14 Write a trainer function for each epoch

- By Paper Author

[72]:
```
def train(args, train_loader, model, criterion, optimizer, epoch):
    model.train()

    total_batches = len(train_loader)
    pbar = enumerate(train_loader)
    pbar = tqdm(pbar, total=total_batches, bar_format='{l_bar}{bar:10}{r_bar}')
    j = 0
    avg_train_loss = 0
    for i, (_, _, input, target) in pbar:
        if args["onGPU"] == True:
            input = input.cuda().float() / 255.0
        output = model(input)

        # target=target.cuda()
        optimizer.zero_grad()

        focal_loss,tversky_loss,loss = criterion(output,target)
        avg_train_loss += loss.item()

        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        pbar.set_description(('%13s' * 1 + '%13.4g' * 3) %
                                    (f'{epoch}/{args["max_epochs"] - 1}',␣
↪tversky_loss, focal_loss, loss.item()))
        j += 1
    return avg_train_loss/j, loss.item()
```

0.15 Train the model with custom data and also print the loss

- This loss is based on the paper

[73]:
```
print("----------------------------------------")
training_loss_last_batch = []
validation_loss_last_batch = []
for epoch in range(0, args["max_epochs"]):
    print(f"Epoch: {epoch + 1}/{args['max_epochs']}")
    poly_lr_scheduler(args, optimizer, epoch)
    for param_group in optimizer.param_groups:
        lr = param_group['lr']
    print("Learning rate: " +  str(lr))
    print()

    # train for one epoch
    model.train()
    avg_train_loss, loss_for_last_batch_train = train( args, train_dataloader,␣
 ↪model, criteria, optimizer, epoch)
    model.eval()

    avg_val_loss = 0
    i = 0
    for batch in val_dataloader:
        _, _, input, target = batch
        if args["onGPU"] == True:
            input = input.cuda().float() / 255.0
        output = model(input)
        focal_loss, tversky_loss, loss = criteria(output, target)
        avg_val_loss += loss.item()
        i += 1

    print()
    print(f"Average Training Loss: {avg_train_loss}")
    print(f"Average Validation Loss: {avg_val_loss/i}")
    print()
    print(f"Training loss for last batch: {loss_for_last_batch_train}")
    print(f"Validation loss for last batch: {loss.item()}")
    print("----------------------------------------")
    training_loss_last_batch.append(loss_for_last_batch_train)
    validation_loss_last_batch.append(loss.item())
```

```
----------------------------------------
Epoch: 1/8
Learning rate: 0.0005

          0/7       0.1699       0.1075       0.2774: 100%|      | 20/20
[00:09<00:00,  2.19it/s]

Average Training Loss: 0.37254337817430494
```

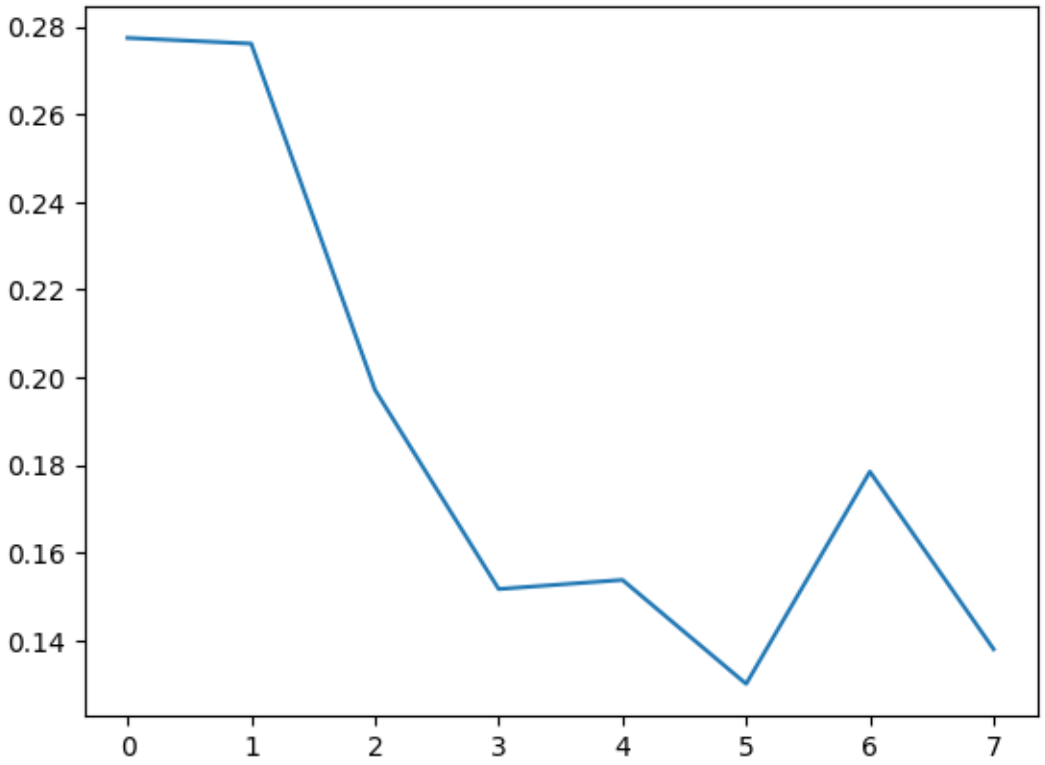
```
Average Validation Loss: 0.3985534608364105

Training loss for last batch: 0.2773999869823456
Validation loss for last batch: 0.3867541253566742
----------------------------------------
Epoch: 2/8
Learning rate: 0.00038281


          1/7       0.2103      0.06577       0.2761: 100%|      | 20/20
[00:09<00:00,  2.19it/s]


Average Training Loss: 0.27785795703530314
Average Validation Loss: 0.27806347608566284

Training loss for last batch: 0.2760956585407257
Validation loss for last batch: 0.2722281217575073
----------------------------------------
Epoch: 3/8
Learning rate: 0.00028125


          2/7       0.1389      0.05824       0.1971: 100%|      | 20/20
[00:08<00:00,  2.36it/s]


Average Training Loss: 0.2262256920337677
Average Validation Loss: 0.25204700728257495

Training loss for last batch: 0.1971188187599182
Validation loss for last batch: 0.3023596405982971
----------------------------------------
Epoch: 4/8
Learning rate: 0.00019531


          3/7       0.1078      0.04385       0.1517: 100%|      | 20/20
[00:09<00:00,  2.17it/s]


Average Training Loss: 0.19270427376031876
Average Validation Loss: 0.23231724401315054

Training loss for last batch: 0.15166451036930084
Validation loss for last batch: 0.2639728784561157
----------------------------------------
Epoch: 5/8
Learning rate: 0.000125
```

```
          4/7      0.09583      0.05792       0.1538: 100%|      | 20/20
[00:09<00:00,  2.17it/s]


Average Training Loss: 0.16028463132679463
Average Validation Loss: 0.19568767150243124

Training loss for last batch: 0.15375079214572906
Validation loss for last batch: 0.24625156819820404
----------------------------------------
Epoch: 6/8
Learning rate: 7.031e-05


          5/7      0.08385      0.04618         0.13: 100%|      | 20/20
[00:08<00:00,  2.38it/s]


Average Training Loss: 0.1617955170571804
Average Validation Loss: 0.19566503167152405

Training loss for last batch: 0.1300331950187683
Validation loss for last batch: 0.16308508813381195
----------------------------------------
Epoch: 7/8
Learning rate: 3.125e-05


          6/7       0.1078      0.07069       0.1784: 100%|      | 20/20
[00:08<00:00,  2.25it/s]


Average Training Loss: 0.15240405797958373
Average Validation Loss: 0.18688194453716278

Training loss for last batch: 0.17844918370246887
Validation loss for last batch: 0.15327155590057373
----------------------------------------
Epoch: 8/8
Learning rate: 7.81e-06


          7/7      0.08908      0.04886       0.1379: 100%|      | 20/20
[00:09<00:00,  2.14it/s]


Average Training Loss: 0.15412542335689067
Average Validation Loss: 0.18836524585882822

Training loss for last batch: 0.13794226944446564
Validation loss for last batch: 0.2271018773317337
```

---

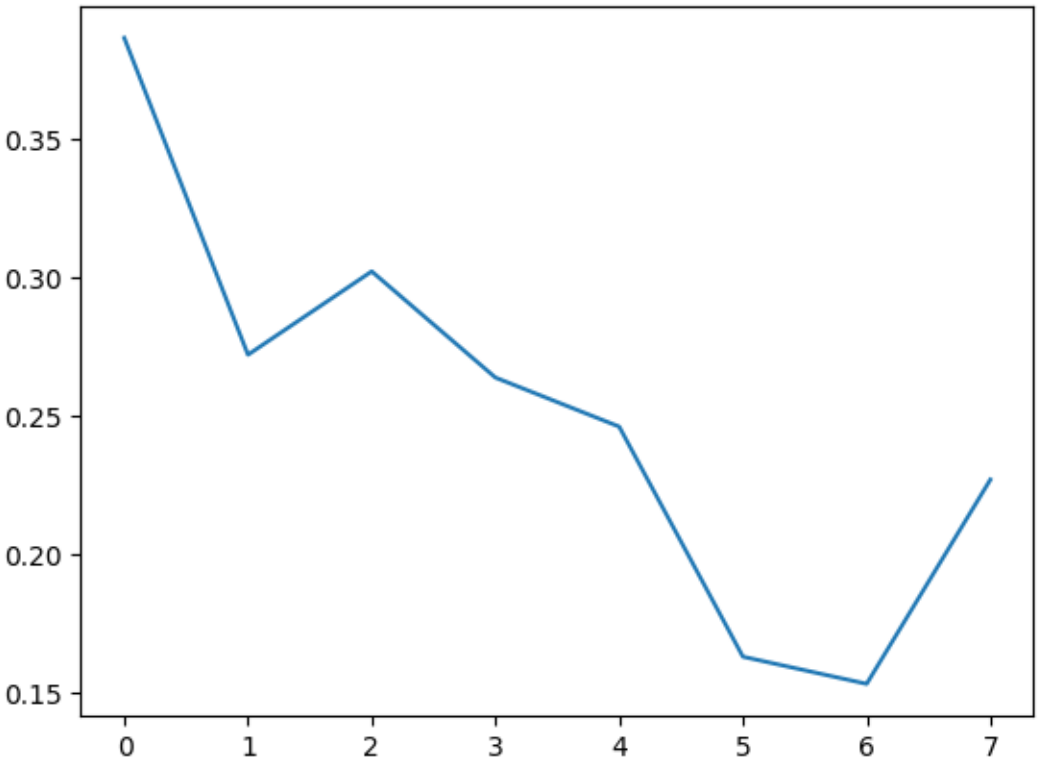
[74]:
```
%matplotlib inline
import matplotlib.pyplot as plt

x = list(range(len(training_loss_last_batch)))
y = training_loss_last_batch

plt.plot(x, y)
```

[74]: [<matplotlib.lines.Line2D at 0x7a8a4675ed40>]



[75]:
```
%matplotlib inline
import matplotlib.pyplot as plt

x = list(range(len(validation_loss_last_batch)))
y = validation_loss_last_batch

plt.plot(x, y)
```

[75]: [<matplotlib.lines.Line2D at 0x7a8b844c2740>]



0.16 Calculating loss on Test data

[76]:
```
avg_test_loss = 0
i = 0
for batch in test_dataloader:
    _, _, input, target = batch
    if args["onGPU"] == True:
        input = input.cuda().float() / 255.0
    output = model(input)
    focal_loss, tversky_loss, loss = criteria(output, target)
    avg_test_loss += loss.item()
    i += 1

print("-----------------------------------------")
print(f"Average Testing Loss: {avg_test_loss/i}")
print(f"Testing loss for last batch: {loss.item()}")
print("-----------------------------------------")
```

```
-----------------------------------------
Average Testing Loss: 0.22908470531304678
Testing loss for last batch: 0.16646476089954376
-----------------------------------------
```

1 Defining functions to calculate Pixel Accuracy and Intersection of Union

- by paper author

[77]:
```
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))
```

[78]:
```
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
```

[79]:
```
@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
```

2 Evaluating metrics

[80]:
```
model.eval()
example = torch.rand(1, 3, 360, 640).cuda()
model = torch.jit.trace(model, example)
da_segment_results,ll_segment_results = val(test_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
```

```
100%|      | 3/3 [00:04<00:00,  1.57s/it]
```

[81]:
```
print(msg)
```

```
Driving area Segment: Acc(0.959)    IOU (0.748)    mIOU(0.851)
Lane line Segment: Acc(0.984)    IOU (0.197)  mIOU(0.591)
```

3 Metrics

- Evaluation metrics are pixel accuracy and IoU(Intersection over Union).
- We have achieved an accuracy of 95.9% for Driving area segment
- We have achieved an accuracy of 98.4% for Lane Line segment.
- An average of 97.15 % pixel accuracy is achieved which is comparable to the original model's accuracy.

[ ]: