**Crack Detection**

**Pre-requisite. Your system should have a GPU with CUDA and CUDNN installed.**

**Step1: Install the packages**

!pip3 install -U pip

!pip3 install -U setuptools wheel

!pip3 install autogluon

Reference visit <https://auto.gluon.ai/stable/install.html>

**Step 2: Download data**

Download data from link: <https://drive.google.com/file/d/1YynE4aZTxJmMMS-vLeQnY9mewjULGHBs/view?usp=sharing>

**Note:** You can use any IDE of your preference to run the lines of code bellow

**Step3: Import Packages**

import autogluon.core as ag

from autogluon.vision import ImagePredictor, ImageDataset

import pandas as pd

import numpy as np

from matplotlib import pyplot as plt

from glob import glob

**Step4: Set path for train and test data**

train\_crack=glob('pavement\_detection/train/crack/\*.jpg')\

+glob('pavement\_detection/train/massive\_crack/\*.jpg')\

+glob('pavement\_detection/train/longitudinal\_crack/\*.jpg')\

+glob('pavement\_detection/train/transverse\_crack/\*.jpg')

train\_nocrack=glob('pavement\_detection/train/normal/\*.jpg')

len(train\_crack),len(train\_nocrack)

test\_crack=glob('pavement\_detection/test/crack/\*.jpg')\

+glob('pavement\_detection/test/massive\_crack/\*.jpg')\

+glob('pavement\_detection/test/longitudinal\_crack/\*.jpg')\

+glob('pavement+detection/test/transverse\_crack/\*.jpg')

test\_nocrack=glob('pavement\_detection/test/normal/\*.jpg')

len(test\_crack),len(test\_nocrack)

**Step5: Create dataframe for train and test**

train\_df=pd.concat([pd.DataFrame(train\_crack),pd.DataFrame(train\_nocrack)])

train\_df['label']=['crack']\*len(train\_crack)+['nocrack']\*len(train\_nocrack)

train\_df.columns=['image','label']

test\_df=pd.concat([pd.DataFrame(test\_crack),pd.DataFrame(test\_nocrack)])

test\_df['label']=['crack']\*len(test\_crack)+['nocrack']\*len(test\_nocrack)

test\_df.columns=['image','label']

**step6: Train model**

predictor = ImagePredictor()

predictor.fit(train\_df, hyperparameters={'epochs': 10})

**step7: Test model**

test\_result = predictor.predict(test\_df)

**step8: Result**

from sklearn.metrics import classification\_report,confusion\_matrix,ConfusionMatrixDisplay

print(classification\_report(test\_df.label,test\_result))

cm=confusion\_matrix(test\_df.label,test\_result)

disp=ConfusionMatrixDisplay(confusion\_matrix=cm,display\_labels=['crack', 'nocrack'])

disp.plot()

plt.show()