Prediction Of Remaining Useful Life of Road from it's Image

by Dhenuvakonda Sai Naveen

1. Introduction

1.1 Overview

This project aims to predict the Remaining Useful Life (RUL) of a road or pavement using image analysis. Users can upload road images, and the system provides an RUL estimate. While data limitations affect accuracy, the project leverages insights from research papers to enhance its methodology.

The chosen approach relies on the Pavement Condition Index (PCI) as a key metric. PCI quantifies road quality by assessing damage severity and density, elements detected and measured by the model. Future plans involve refining accuracy with improved data sources, ensuring more precise RUL predictions and aiding road maintenance.

I have taken input as images of road sections, detected and assessed severity of the types of damages to calculate the Pavement Condition Index and used that to predict RUL and deployed the entirety of it on Streamlit. All code pushed to GitHub Repo link and Streamlit app link.

1.2 Industrial Use Case

In the realm of road maintenance, this project bears immense significance. Each year, substantial resources are expended on roads that are already in good condition, while many deteriorating roads remain unattended. Such inefficiencies impose heavy costs on both companies and governments. The emergence of a tool capable of providing estimations of a road's remaining usable life (RUL) and identifying the timing for necessary maintenance addresses these issues. This predictive capability offers the potential for resource allocation optimization.

One of the key advantages lies in the simplicity of visual assessment. Merely requiring images of road segments, this approach can be readily facilitated through readily accessible sources, such as Google Street View. With further refinement, this project holds the promise of integration into existing road quality assessment software systems.

2. Existing Work

Prediction of the remaining service life (RSL) of pavement is a challenging task for road maintenance and transportation engineering. The prediction of the RSL estimates the time that a major repair or reconstruction becomes essential. The conventional approach to predict RSL involves using non-destructive tests. These tests, in addition to being costly, interfere with traffic flow and compromise operational safety. In this paper, surface distresses of pavement are used to estimate the RSL to address the aforementioned challenges. To implement the proposed theory, 105 flexible pavement segments are considered. For each pavement segment, the type, severity, and extent of surface damage and the pavement condition index (PCI) were determined. The pavement RSL was then estimated using non-destructive tests include falling weight deflectometer (FWD) and ground-penetrating radar (GPR). After completing the dataset, the modelling was conducted to predict RSL using three techniques include support vector regression (SVR), support vector regression optimized by the fruit fly optimization algorithm (SVR-FOA), and gene expression programming (GEP). All three techniques estimated the RSL of the pavement by selecting the PCI as input. The correlation coefficient (CC), Nash–Sutcliffe efficiency (NSE), scattered index (SI), and Willmott's index of agreement (WI) criteria were used to examine the performance of the three techniques adopted in this study. In the end, it was found that GEP with values of 0.874, 0.598, 0.601, and 0.807 for CC, SI, NSE, and WI criteria, respectively, had the highest accuracy in predicting the RSL of pavement. This has been taken from the Paper titled Comparative Analysis of Machine Learning Models for Prediction of Remaining Service Life of Flexible Pavement.

3. Approach and Workflow adopted

3.1 Approach

I had been going through various papers to determine the correct or one viable approach to come up with a solution for this use-case. I have summarized my findings here. The most common way to estimate RUL is to find one metric related to the road like PCI, IRI or DMI in a timely manner as in collect its data over a period of time and extrapolate it to find when the index value will fall to one particular range of values. Getting such data was not possible as none of it was open source and wasn't readily available. There were some datasets with historical data on PCI, but either there were no supporting parameters or there wasn't data pertaining to one location which was needed. So this struggle for data was one of the major reasons for the inaccuracy of this project. Most of this sort of data is private and closed, not open source.

My basic idea was to use image data which was quite rare. Then get data on damages from them and then use that to calculate PCI which in turn has a direct correlation to RUL. I have used YOLOv8 in order for object detection. This is further discussed in Workflow section below.

3.2 Workflow

Major components and the workflow of this project is divided as follows:

(a) Getting Image Data
(b) Selecting Model and Training and Inference of Damage detection model
(c) Collecting data on How to calculate PCI from detected damages
(d) Historical data on PCI and related data to estimate RUL (Machine Learning Approach)
(e) Finding accurate formulas to get RUL (Algorithmic approach).

Let us discuss one by one.

(a) Getting Image Data

In order to get data for detecting and classifying road damages, I needed a lot of data for my model to learn and accurately predict. It was quite a challenge to find suitable data but I was lucky enough to find the RDD2022 Dataset, Road Damage Dataset, RDD2022, which comprises 47,420 road images from six countries, Japan, India, the Czech Republic, Norway, the United States, and China. The images have been annotated with more than 55,000 instances of road damage. Four types of road damage, namely longitudinal cracks, transverse cracks, alligator cracks, and potholes, are captured in the dataset. The annotated dataset is envisioned for developing deep learning-based methods to detect and classify road damage automatically. The dataset has been released as a part of the Crowd sensing-based Road Damage Detection Challenge (CRDDC2022). This data has been quite good in getting a good detection model.

```
Folder PATH listing for volume Windows
Volume serial number is 520B-286B
C:.
\---RDD2022
    +---China_Drone
    |   \---train
    |       +---annotations
    |       |   \---xmls
    |       \---images
    +---China_MotorBike
    |   +---test
    |   |   \---images
    |   \---train
    |       +---annotations
    |       |   \---xmls
    |       \---images
    +---Czech
    |   +---test
    |   |   \---images
    |   \---train
    |       +---annotations
    |       |   \---xmls
    |       \---images
    +---India
    |   +---test
    |   |   \---images
    |   \---train
    |       +---annotations
    |       |   \---xmls
    |       \---images
    +---Japan
    |   +---test
    |   |   \---images
    |   \---train
    |       +---annotations
    |       |   \---xmls
    |       \---images
    +---Norway
    |   +---test
    |   |   \---images
    |   \---train
    |       +---annotations
    |       |   \---xmls
    |       \---images
    \---United_States
        +---test
        |   \---images
        \---train
            +---annotations
            |   \---xmls
            \---images
```

This is the structure of the data which I had accessed from their repository https://github.com/sekilab/RoadDamageDetector.

In order to train the model, I had to work on the directories part of it quite intensely. Firstly, all images had to be in one folder for any model as that is the standard. All the annotations were in PASCAL VOC XML format which had to be converted into YOLO label format which required quite a lot of python scripting.

```xml
<annotation>
	<folder>images</folder>
	<filename>China_Drone_000000.jpg</filename>
	<size>
		<width>512</width>
		<height>512</height>
		<depth>3</depth>
	</size>
	<segmented>0</segmented>
	<object>
		<name>D10</name>
		<pose>Unspecified</pose>
		<truncated>0</truncated>
		<difficult>0</difficult>
		<bndbox>
			<xmin>323</xmin>
			<ymin>213</ymin>
			<xmax>390</xmax>
			<ymax>245</ymax>
		</bndbox>
	</object>
</annotation>
```

PASCAL VOC XML FORMAT

```
0 0.6962890625 0.447265625 0.130859375 0.0625
```

YOLO labels format

all_annotations	23-09-2023 12:58	File folder
all_images	02-10-2023 15:07	File folder
all_labels	23-09-2023 14:50	File folder

After this was done, I was ready for training and inference. But a little bit of thought had to go into Selecting appropriate model as well.

(b) Selecting Model and Training and Inference of Damage detection model

There are quite a number of models available for image processing and object detection/segmentation/classification. But according to my requirement, I decided to use YOLO detection model. The latest release is YOLOv8 which is what I have used. I fine tuned the model onto my dataset and was able to get quite good results. I used the medium sized model in YOLOv8 series as my GPU was only capable of handling such a model. (Nvidia GTX 1660ti).

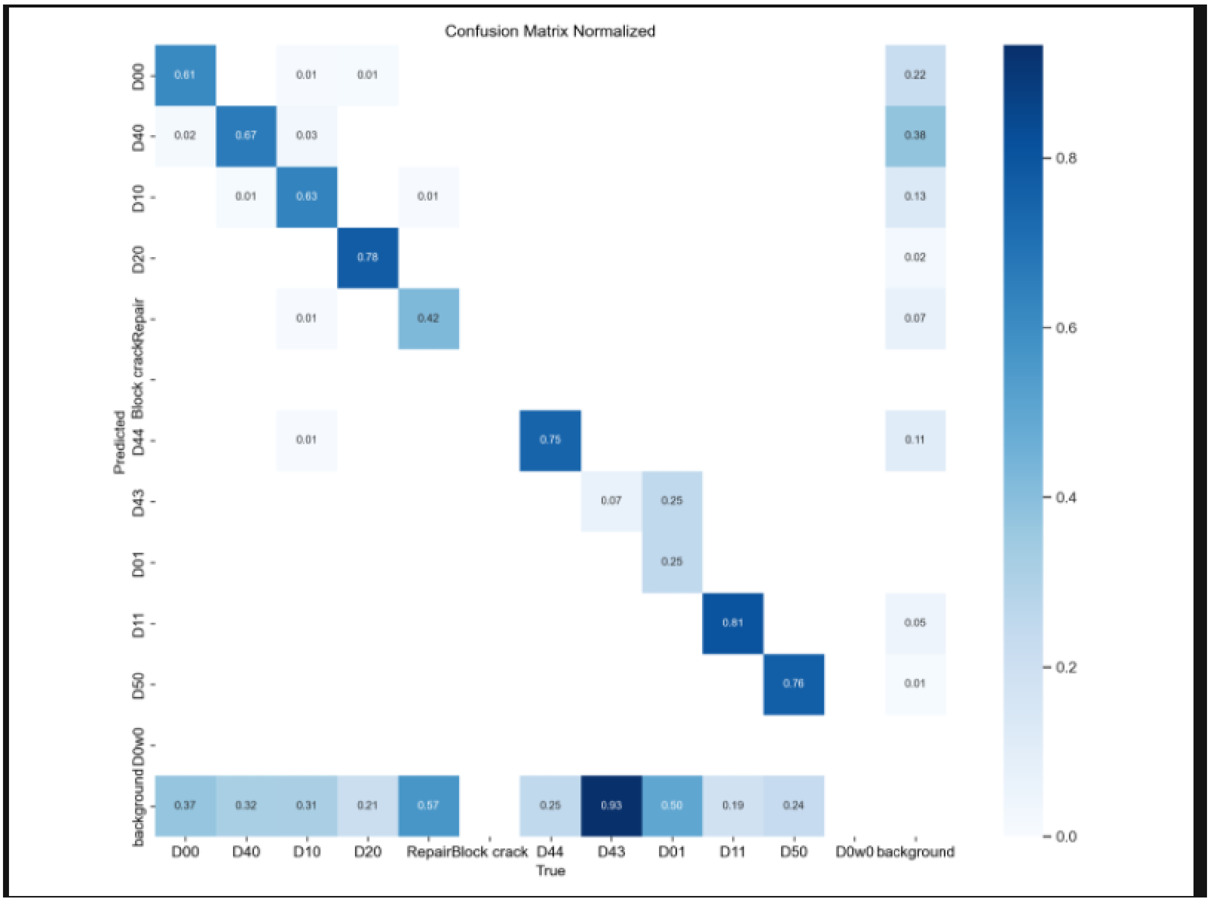
Some hyperparameters I had to first look into

- Input Image Size: Determines the dimensions of input images. I used the standard 640p size.
- Learning Rate: Affects training speed and convergence. Was default value.
- Number of Classes: Total object classes for detection. I had 11, but did not condense it down as the class were quite differentiable.

- Anchor Boxes: Shapes and sizes for object prediction. Did not use as I figured it was not going to drastically effect accuracy in output.
- Backbone Architecture: Choice of feature extraction network. I decided to use the pretrained model which was trained on COCO dataset. I used the best weights for further predictions
- Epochs: 100
- Patience: This was an important parameter. It was set to 50. This meant if there was no increase or decrease in Loss values over 50 epochs, the model would stop training to save resources and time.

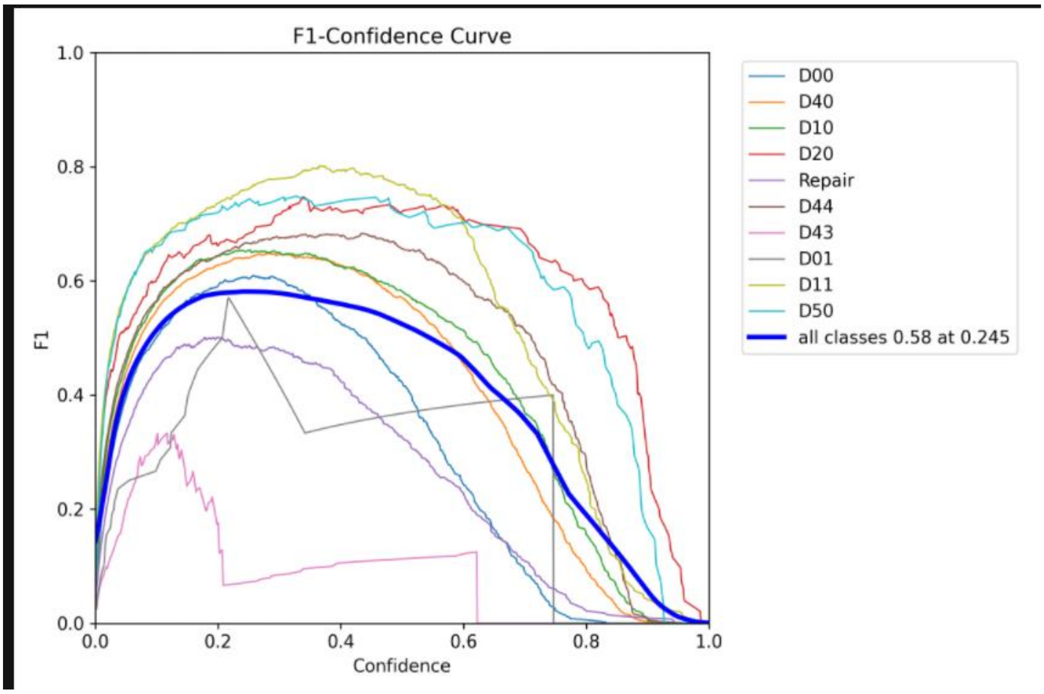
Once training was complete, these are the results it generated.

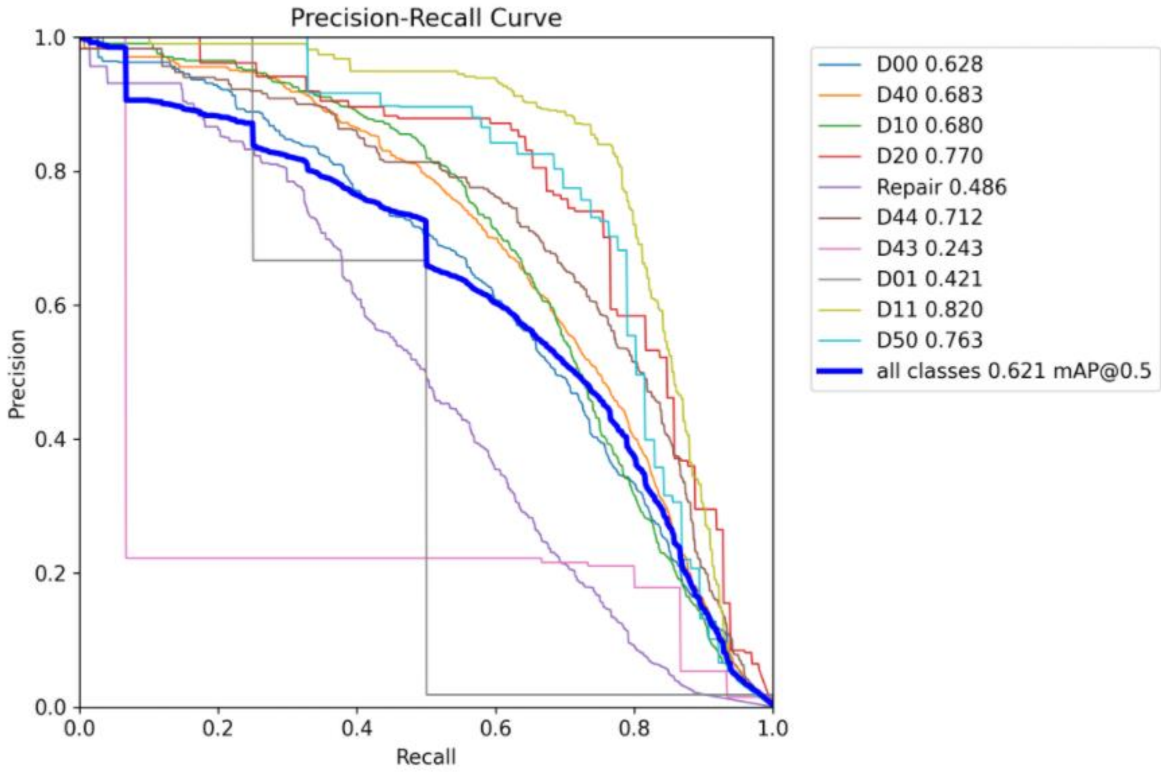
<u>Confusion Matrix between the Classes</u>



<u>The F1 curve.</u>

F1 score is a machine learning evaluation metric that measures a model's accuracy. It combines the precision and recall scores of a model. The accuracy metric computes how many times a model made a correct prediction across the entire dataset. The curve for its variation throughout training has been provided.



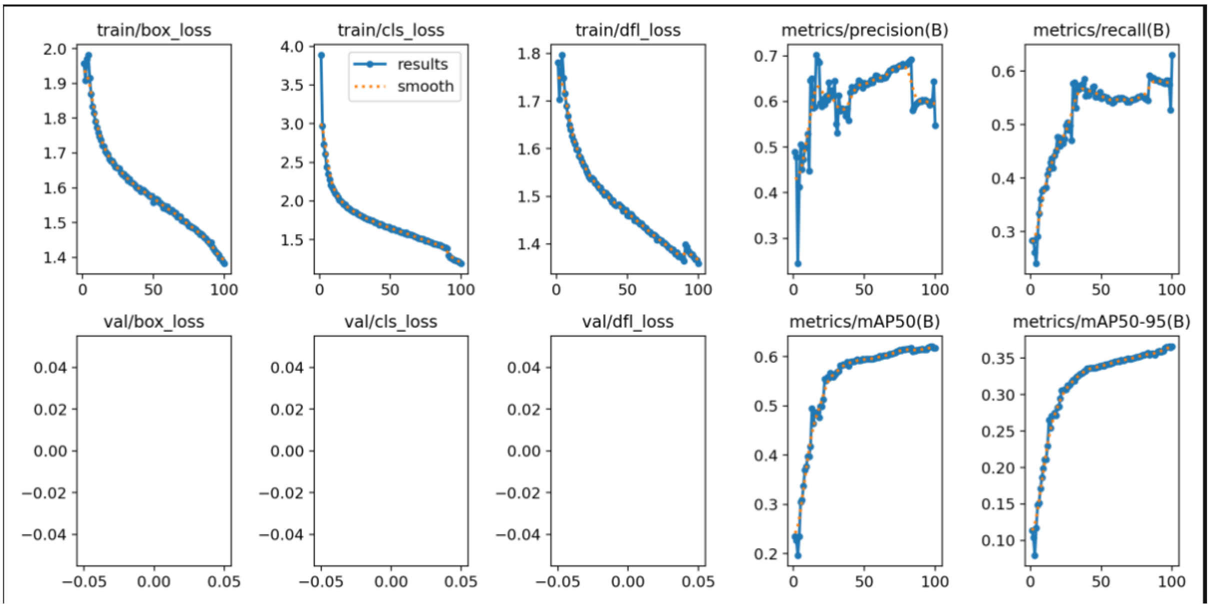


At the end of training, box loss was at 1.382, class loss was at 1.19 and mAP50-95(B) was at 0.36

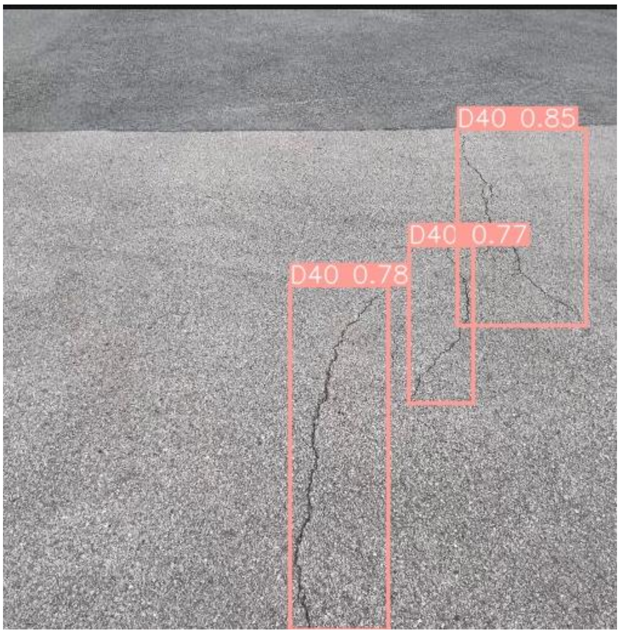
Now what is mAP50-95(B)?

- ➔ means average mAP over different IoU thresholds, from 0.5 to 0.95, step 0.05 (0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95).
- ➔ To evaluate object detection models like R-CNN and YOLO, the mean average precision (mAP) is used. The mAP compares the ground-truth bounding box to the detected box and returns a score. The higher the score, the more accurate the model is in its detections.

- ➔ IoU is Intersection over Union. Intersection over Union (IoU) is a widely-used evaluation metric in object detection and image segmentation tasks. IoU measures the overlap between predicted bounding boxes and ground truth boxes, with scores ranging from 0 to 1.



<u>An example of inference</u>



(c) Collecting data on How to calculate PCI from detected damages

Post training, I had 11 classes detected by the model. Now a way to finding PCI from this information was required. Almost 1.5 weeks of research was put into figuring out the correct and most accurate approach to calculating Pavement Condition Index from the damages detected. It has been explained here.

The roadmap to calculating PCI from damages was

Get damages detected > get width, height and relative area information from results of the model > Roughly estimate the severity and density of distress > Based on what type of crack or damage it is, assign a weight and then calculate total Damage Manifestation Index or Deduct Value > Get PCI from this data.

The damage classes are :

{0: 'D00', 1: 'D40', 2: 'D10', 3: 'D20', 4: 'Repair', 5: 'Block crack', 6: 'D44', 7: 'D43', 8: 'D01', 9: 'D11', 10: 'D50', 11: 'D0w0'}

Different types of damages has been shown below:

Damage Type			Detail	Class Name
Crack	Linear Crack	Longitudinal	Wheel mark part	D00
			Construction joint part	D01
		Lateral	Equal interval	D10
			Construction joint part	D11
	Alligator Crack		Partial pavement, overall pavement	D20
Other Corruption			Rutting, bump, pothole, separation	D40
			White line blur	D43
			Cross walk blur	D44

Source: Road Maintenance and Repair Guidebook 2013 Design and Subcommittee (2013) in Jap

researchgate.net

The first step was to get information on what type of damage it is. Now for the sake of simplifying calculations, I decided to combine classes on the basis of their similarity. They are Linear Cracks, Alligator Cracks and Others. D00,D01,D10,D11 are Linear Crack,D20 is Alligator Crack and D40,D43,D44 and etc are Others. I did this based off the paper I found on ResearchGate.

Now after going through various research papers, articles, journals and reports, I found a good weight to damage class value which I normalized to account for the condensing of classes. The weights I decided on are

"linear": 2.7,

"alligator": 2.2,

"other": 1.5.

The actual weight values are as seen:

Pavement			Wi	Severity of Distress (Si)					Density of Distress (Di)				
				Very Slight	Slight	Moderate	Severe	Very Severe	Few	Intermittent	Frequent	Extensive	Throughout
									<10	10-20	20-40	40-80	>80
				1	2	3	4	5	1	2	3	4	5
Surface Defects		Ravelling & loss of surface aggregate	3.0		x					x			
Surface Defects		Flushing	1.5		x					x			
Surface Deformations		Rippling and Shoving	1.0	x					x				
Surface Deformations		Wheel Track Rutting	3.0	x					x				
Surface Deformations		Distortion	3.0		x					x			
Cracking	Longitudinal Wheel Track	Single and Multiple	1.5	x					x				
Cracking	Longitudinal Wheel Track	Alligator	3.0	x					x				
Cracking	Centerline	Single and Multiple	0.5			x					x		
Cracking	Centerline	Alligator	2.0	x					x				
Cracking	Pavement Edge	Single and Multiple	0.5			x					x		
Cracking	Pavement Edge	Alligator	1.5			x					x		
Cracking	Transverse	Half, full and multiple	1.0	x					x				
Cracking	Transverse	Alligator	3.0	x					x				
Cracking	Longitudinal – meander or mid-lane		1.0		x					x			
Cracking	Random		0.5										

OGRAS MILESTONES • V9#4 • DECEMBER 2009

Now I had all these values ready, Now I needed the total deduct value which subtracted from 100 would give me the PCI value.

Equation 6.2

$$PCI_i = PCI_{max} - \Sigma \text{ Deduct}$$

Where

PCI_i = individual condition index based on measured condition 1

PCI_{max} = value for perfect condition with no measured defects

Deduct = deduct value assigned to distress type, severity & extent

Here, PCI max is 100 as that is the highest index value possible

So in order to calculate Severity and Density, I used the width and height of the bounding box for severity and compared the area of the bounding box and the total image area to get density. Then I ran inference and noted down the readings of various types and scale of damages. Then came up with a scale to rate these both. The scale is as follows

Distress Params

- Severity Of Distress-> Average of width and height is taken to get a rough estimate of length of the crack which points towards severity roughly.

If this average is

<500: Very Slight(1) ,

<800: Slight(2),

<1200: Moderate(3)

<1500: Severe,

1500+:Very Severe(5)

- Density Of Distress-> Relative area of the bounding box and the total area of the image itself,can be improved with a segmentation model

If relative area:

1-3 : Few(1)

4-6 : Intermitent(2)

7-10 : Frequent(3)

11-14 : Extensive(4)

15+ : Throughout(5)

Then the weight and sum of these ratings is multiplied to get deduct value, then all such values are added and finally subtracted from max PCI i.e 100 to get PCI Value.

Table 3. form for road pavement conditions at STA 0 +300 to 0 + 400

FIELD OBSERVATION DATA FOR EACH UNIT SAMPLE

Flexible Pavement

Road : Mandarin - Bundarang Nicolao Lobato and Bebornik	Date : 20/06/2021
	Station : 0 + 300 s/d 0 + 400
Surveyed by : Brigida A. F. De Almeida	Unit Sample : 4
JENIS KERUSAKAN	Wide Area : 500.00 m^2

1. Alligator cracking	11. Patching	100 m
2. Bleeding	12. Polished	
3. Block cracking	13. Ravelling/Weathering	
4. Corrugation	14. Rutting	
5. Depression	15. Shoving From PCC	
6. JT. reflection	16. Swell	
7. Longs & Trans Cracking	17. Bump and Sag	
8. Pothole	18. Slippage Cracks	5 m
9. Lane/Shoulder drop-off	19. Edge Cracking	
10. Polished Aggregate	20. Delamination	

TYPE OF DAMAGE AVAILABLE

		7			19			4			3		
		8	x	0.8	100	x	0.3	20	x	2	9	x	1
Total Severity	L										9.0	m^2	
	M	6.4	m^2		30	m^2							
	H							40	m^2				

PCI CALCULATION

DISTRESS TYPE	SEVERITY	DENSITY %	DEDUCT VALUE
7	M	1.28	4
19	M	6.0	11
4	H	8.0	59
3	L	1.8	2
TOTAL DEDUCT VALUE (TDV)			76
CORRECTED DEDUCT VALUE (CDV)			65

PCI = 100 - *CDV*
= 100 – 65
= 35

Rating = Poor

So in summary, To determine the Pavement Condition Index (PCI) for a road section, follow these steps:

- Assess Distresses Inspect the road section for various types of distresses, noting their severity, density, and deduct value.

- Deduct Value Calculation: Calculate the deduct value for each distress by finding a rating for its density and severity and rate it on a standardized scale.

- Total Deduct Value: Sum all the individual deduct values to obtain the total deduct value for the road section.

- Calculate PCI: Calculate the PCI by subtracting the Corrected Deduct Value (CDV) from 100 using the formula: PCI = 100 - CDV.

- PCI Rating: Based on the calculated PCI value, assign a rating to the road section using the following ranges:

  - 85-100: Excellent
  - 70-85: Very Good
  - 55-70: Good
  - 40-55: Fair
  - 25-40: Poor
  - 10-25: Very Poor
  - 0-10: Failed

These steps help assess the condition of the road section and categorize it into one of the predefined PCI ratings, aiding in road maintenance and rehabilitation decisions.

(d) Historical data on PCI and related data to estimate RUL (Machine Learning Approach)

I failed to get an accurate prediction of Remaining Useful Life from this approach and in the future plan to get better data and develop a model rather than relying on algorithmic approach.

My plan was to collect historical PCI values data along with location details so that I can get weather and traffic data, develop it into a Time Series dataset and train a model like SARIMAX of fbprophet to estimate in which year the PCI value would hit a particular value. I did find data like this but the issue was only 4 years worth of data was available for a particular location which was not enough for a model to pick up trends. The streets were all next to each other as well so the weather and traffic data was quite similar which made it even harder to get a good value.

I picked this data up from “Pavement condition for provincial highways” from open Canada data website. It had data from 2014 to 2017 on the following parameters

Section ID	Highway	DIR	LHRS	Offset	FROM_Distance	TO_Distance	PCI	DMI	IRI	Pave_Type	Pavement_Section_From	Pavement_Section_To	FUNC_CLASS

I had data like this for 4 years.

How did I develop a time series dataset from these datasets?

- ➔ I added a column to indicate what year it is so 2014 dataset had one parameter with just one datetime type value that was 2014 and so on.
- ➔ Since only Pavement Section From and To was given, I decided to geocode this to find the latitude and longitude, I used geopy library and Bing API to do so.
- ➔ After finding geolocation, I used this data and used MeteoStat API to get weather data.
- ➔ Finally, I combined this data and did some cleaning and feature engineering to improve it’s quality
- ➔ I tried using SARIMAX, ARIMAX, Extrapolate, fbprophet, Random Forest Regressor and many more models but none of them were able to catch proper trends between them due to reasons I have already explained.

(e) Finding accurate formulas to get RUL (Algorithmic approach)

Since I was unable to develop a model to predict RUL, I decided to do further research and figure out another approach to calculate Remaining Useful Life from PCI as I was able to calculate PCI.

There were already quite a lot of equations linking RUL to various other indices

Table 1. Models for the prediction of RSL [**26**,**27**,**28**,**29**,**30**,**31**,**32**,**33**].

Category	Model Inputs	Equation	Author
1. Based on pavement responses	ε_t = Tensile strain at the bottom of the asphalt layer, E_1 = Elastic modulus of asphalt, f_1, f_2, and f_3 = Regression coefficients.	$RSL_{fatigue} = f_1(\varepsilon_t)^{-f_2}(E_1)^{-f_3}$	Huang (1993)
	ε_c = Compressive strain at the top of the subgrade, f_4 and f_5 = Regression coefficients.	$RSL_{rutting} = f_4(\varepsilon_c)^{-f_5}$	Huang (1993)
	ε_t = Tensile strain at the asphalt layer bottom, M_R = Resilient modulus.	$RSL_{fatigue} = 0.1001\,(\varepsilon_t) - 3.565(M_R)^{-1.4747}$	Das & Pandey (1999)
	ε_r = Horizontal tensile strain at the bottom of the asphalt layer, E_{AC} = Modulus of asphalt, a, b, and c = Constant coefficients of regression.	$ln\left(RSL_{fatigue}\right) = a - b\,ln(\varepsilon_r) - c\,ln(E_{AC})$	Hossain & Wu (2002)
	ε_t = Tensile strain at the asphalt layer bottom, K and c = Regression coefficients.	$RSL_{fatigue} = K(\varepsilon_t)^{-C}$	Park & Kim (2003)
2. Based on pavement quality indices	IRI = International roughness index, a = Initial IRI (where age is zero), b = Curvature of performance line.	$RSL = \frac{ln\left(\frac{IRI_{terminal}}{a}\right)}{b} - (Current\ age)$	Al-suleiman & Shiyab (2003)
	PCI = Pavement Condition Index.	$RSL = 4.1872\,ln\,(PCI) - 14.728$	Setyawan et al. (2015)
3. Based on the result of the non-destructive test	δ = Pavement surface curvature, $\delta = D_0 - D_{20}$, α and β = Material constants.	$RSL_{fatigue} = \alpha\left(\frac{1}{0.0023\delta+0.00002}\right)^{\beta}$	Saleh (2016)
	AUPP = Area under pavement profile, $AUPP = \frac{5D_0-2D_{30}-2D_{60}-D_{90}}{2}$, α and β = Material constants.	$RSL_{fatigue} = \alpha\left(\frac{1}{0.0000023AUPP^{0.912}}\right)^{\beta}$	Saleh (2016)

* D_i = Deflection of pavement surface on distance i cm from the center of the loading plate in the FWD test.

I was able to find a simple relation between PCI AND RSL which is basically RUL and that was

RUL=4.1872* ln(PCI) – 14.117

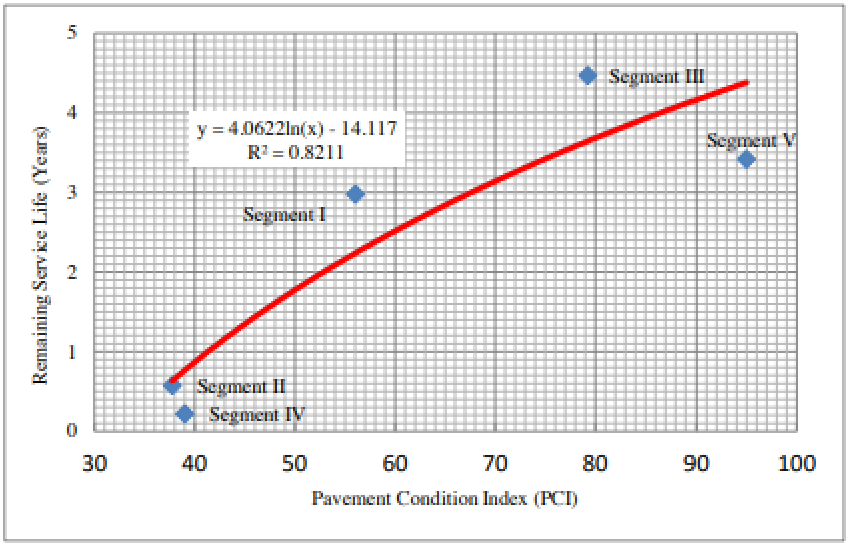


Fig.4 The correlation between PCI value and The Remaining Life Service

I use this formula to calculate RUL and it was quite accurate which I cross verified with other equations involving DMI and other factors. This is what I deducted from Literature review of various research papers.

The Correlation between PCI Values and the Remaining Service Life The data used in this analysis are the value of PCI and the remaining service life calculated based on deflection from FWD measurement. The correlation between the value of PCI and the remaining service life is carried out using the Microsoft Excel program. The pattern of the relationship between the PCI and the remaining service life is presented in Fig. 4. It shows that the group of road segments that have higher values of PCI tend to have longer remaining service life. Instead, the segment group with a lower PCI value also has a tendency of a shorter remaining service life. The relationship between the value of PCI with the remaining service life is $y = 4.1872\ln(x) - 14.728$, where y is the value of remaining service life (years) and x is the value of PCI. The correlation coefficient (r) of this relationship is 0.88 with a strong degree of interpretation. In terms of its relation to the estimated need for treatment and rehabilitation of roads, pavement remaining life prediction based on functional failure is easier than those based on structural failure. PCI value may help to identify the segment, which needs to be conducted a preventive maintenance in order to prevent the further deterioration. Therefore, in this study, the critical limit value of PCI could be determined to select the right time of road handling for segment examined. Based on the critical limitation, the graph is developed to predict the remaining service life based on PCI values. However, further research is required to understand in detail this relationship for better understanding of this correlation.

This brings us to the end of Approach and Workflow for this project. We move on to a brief summary of code which has been stored in a Github Repository.

4.Code Summary and Brief Explanation.

Please access the repository for this project here-> https://github.com/DiazOnFire/Road-RUL-Predictor

Files included are :

1. Defaults: Contains images which are default images that can be used to see output. This has been done in Streamlit and the project has been deployed using Streamlit Cloud.

2. Runs/detect : Has some example results I saved whilst testing the model post training.

3. ImageCalc: This has all the python scripts to extract information from the output gotten from YOLOv8 model, calculating the various values required to calculate PCI and finally the RUL as well. All this has been used in app.py as well to simplify execution.

4. App.py: This is the main file which streamlit executes. Here, we have 3 functions namely output_fetch(), modify() and PCI_Calc(), running the model on the uploaded image and then passing this result through the said functions to finally get Output which is displayed.

5. Best.pt: Anyone can use these weights to run inference on their system as well. These are the best weights attained after training the model on the RDD2022 dataset.

6. dataPrep_2014_15 and dataPrep_2016_2017: These notebooks contain the feature engineering and analysis done on the Canadian Highways dataset mentioned in Workflow, Here Geocoding API and another API for getting weather data has been used.

7. Datamaker: Here, the RDD2022 dataset has been modified at the directory level and it has been split into train test and validate folders as well. PASCAL VOC XML Format to YOLO conversion has been done as well.

8. Getting_weather_data: MeteoStat API has been used and a function has been written to fetch weather information for the 4 years of data.

9. Modeltraining: Different models were used to predict RUL from the 4 year data but I removed it as it didn't fetch good results.

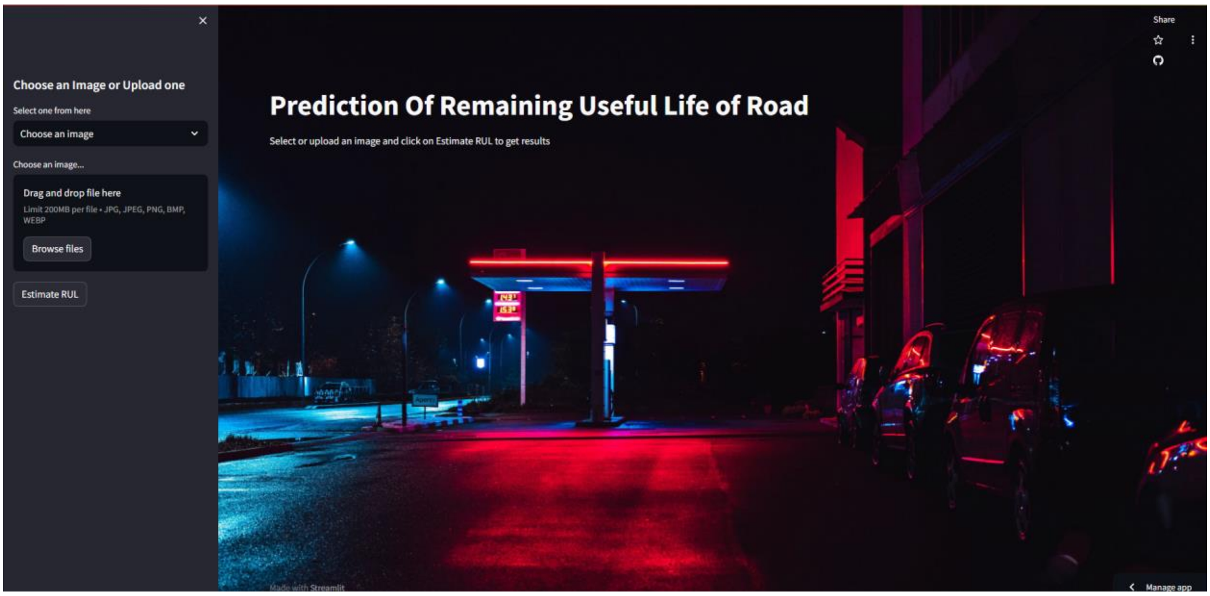
10. Training: YOLOv8 model has been trained and finetuned in this file.

The project has been deployed on Streamlit which can be accessed here

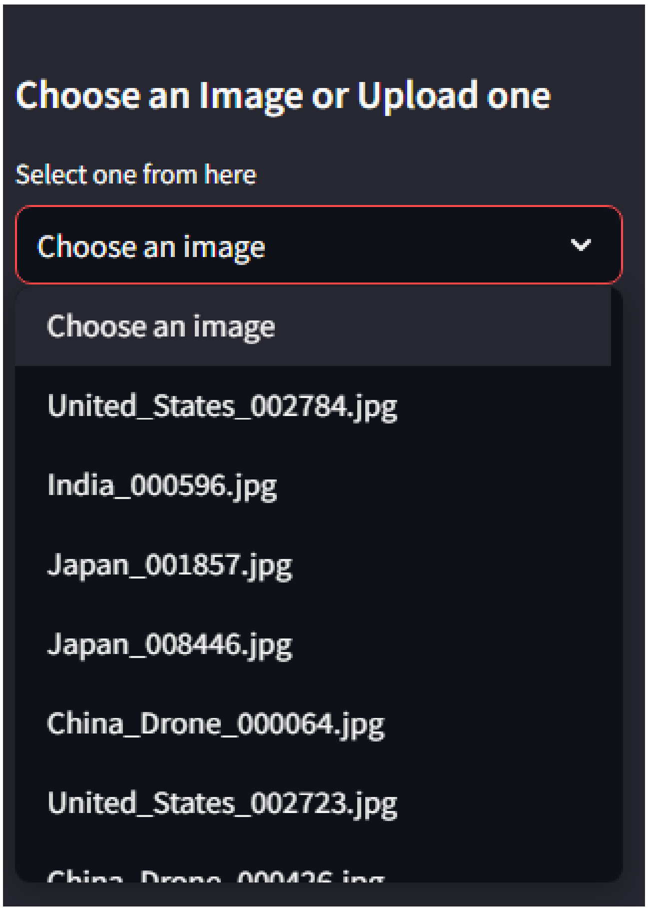
-> https://road-rul-predictor.streamlit.app

5.Results And how to use model

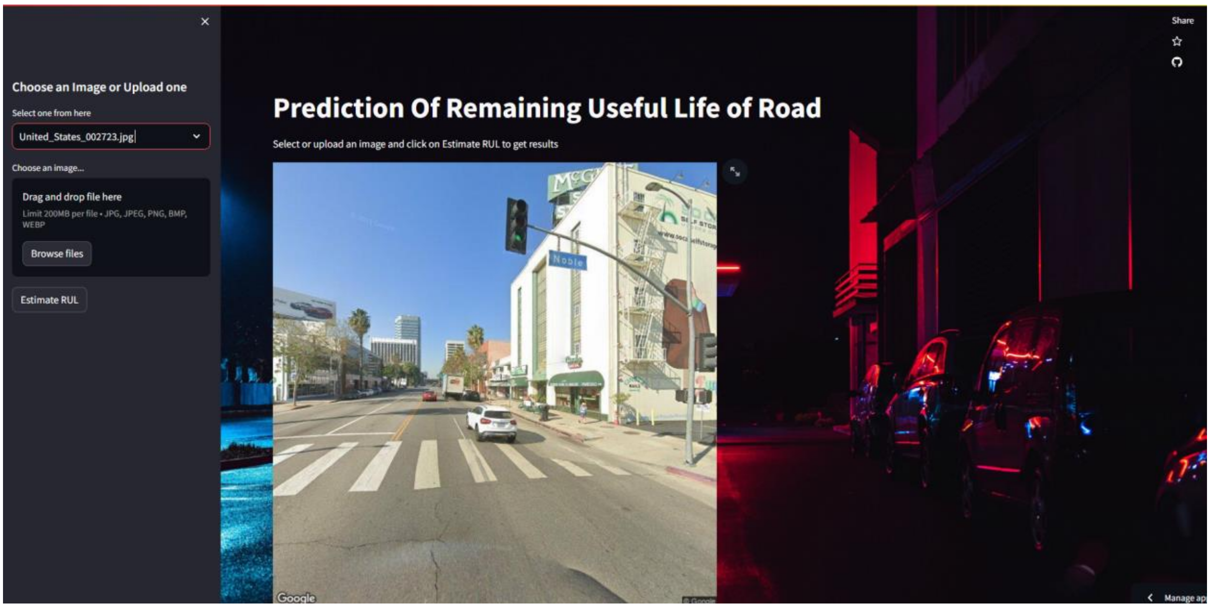
Model accuracies have been mentioned already and I will show how to use the streamlit app to get results.



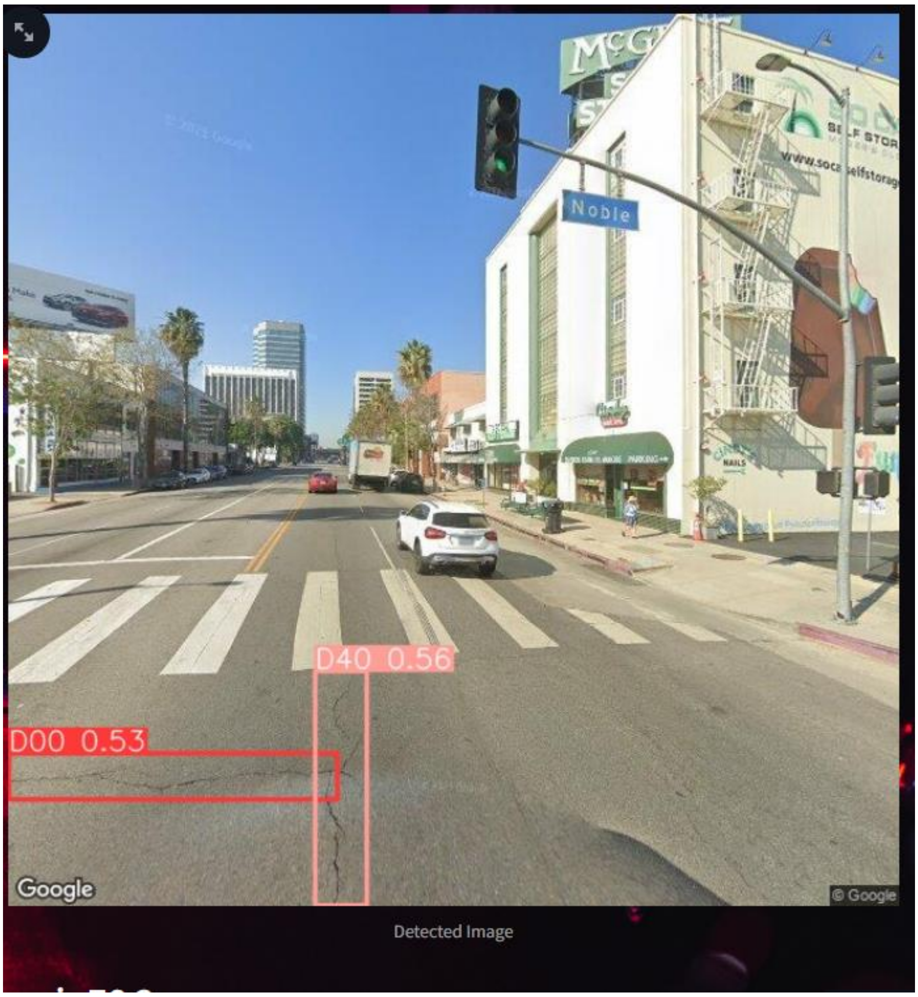
1. Upload an image or just choose one from the drop down menu.



2. You can now see the uploaded image, now to get the RUL estimate, click on Estimate RUL button



3. Now you can see the bounding boxes predicted by the model as well as the PCI value, A description of the quality of the road and the RUL value.



4. You can now use another image if needed to get the results again.

6.Scope for Improvement

<u>Damage Detection Model</u>

Another model can be used instead of YOLO, something like U-NET or a pretrained Fast R-CNN model. The hyperparameters have not been properly finetuned due to limitations in hardware and further fine tuning and improved batch processing can help in improving bounding box predictions. Using a stronger model than yolov8m will also massively improve results but again due to the limitation in GPU memory, I cannot train a larger model.

<u>Severity and Density Assessment.</u>

Rather than relying on bounding box dimensions to calculate severity and density, developing a Segmentation model can help in getting accurate values to represent the above two parameters. An Instance Segmentation model would be preferred as multiple occurrences of the same type of damage can be detected. Using an improved Weight scale can also help as I used quite a basic one and training a model to identify these weights is possible as well but due to lack of Distress values data and its contribution to the Deduct value is not really available, I had to resort to a simple yet powerful methodology.

<u>RUL Calculation</u>

I really wanted to use a time series extrapolation model to get RUL but the data was just not in the correct form for a model to learn any sort of proper trends. Having historical data of one place for over 10 years with monthly values is good enough for such a model. APIs not being free also hindered accurate values for latitude, longitude and weather data. There was no free API to get traffic data as well. TomTom offered all of these services but at an exorbitant rate. I hope to use better APIs and better data to predict RUL not algorithmically but using a trained model

Existing Research

There is quite a lot of research that points towards prediction of RUL using PCI but all other methods mentioned have no data or code backing it up so it is quite hard to get a grip of. Further research and maybe collaboration with local transport authorities or public works departments will help in getting on the ground and getting my own data which can help in getting a highly accurate Service Life value.

I am very happy that I was assigned this work and learnt an incredible number of skills like a workflow to researching about a use case, using various APIs, working with directories using python and using Object detection models as well. This has set a good foundation stone for me to explore the world of Computer Vision and image processing.

7.References

1. Comparative Analysis of Machine Learning Models for Prediction of Remaining Service Life of Flexible Pavement
2. Remaining service life prediction using road structure performance data with pavement condition index (PCI) and Benkelman beam (BB) methods
3. Application of Pavement Condition Index (PCI) on The Assessment of The Kalumata – Fitu Highway Section of Southern of Ternate City
4. Pavement Condition Index (PCI) There's More (and Less) to the Score
5. Introducing mathematical modeling to estimate pavement quality index of flexible pavements based on genetic algorithm and artificial neural networks
6. PAVEMENT CONDITION INDEX 101
7. Pavement Condition Index (PCI) Method for Road Damage Analysis
8. Predicting the remaining service life of road using pavement condition index
9. EVALUATION OF FLEXIBLE PAVEMENT DAMAGE USING THE PCI (PAVEMENT CONDITION INDEX) METHOD BASED ON CORE DRILL DATA, CASE STUDY: ON THE MANDARIN ROAD TO THE PRESIDENT NICOLAO LOBATO ROUNDABOUT AND THE BEBONUK ROAD IN DILI CITY, TIMOR-LESTE"
10. Assessing severity of road cracks using deep learning-based segmentation and detection
11. Pavement condition for provincial highways open Canada dataset website
12. Development of Remaining Service Life (RSL) of pavements for different road networks
13. RDD2020: An Image Dataset for Smartphone-based Road Damage Detection and Classification
14. Predicting the Remaining Service Life of Road Using Pavement Condition Index
15. Remaining Useful Life (RUL) Prediction of Equipment in Production Lines Using Artificial Neural Networks
16. https //github.com/sekilab/RoadDamageDetector.