

CNN BASED RAIL SURFACE CRACK DETECTION USING DIFFERENT MACHINE LEARNING TECHNIQUES

Submitted in partial fulfillment of the requirements
for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

Ravi Kumar Dusi(38110454)
G.RamaSujith(38110447)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119**

MARCH – 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with “A” grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Ravi Kumar DUSI(38110454)** and **G.RAMASUJITH(38110447)** who carried out the project entitled “ **CNN BASED RAIL SURFACE DETECTION USING DIFFERENT MACHINE LEARNING TECHNIQUES**”

under my supervision from November 2021 to March 2022.

Internal Guide

DR. P. JEYANTHI

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **RAVI KUMAR DUSI (Reg No:38110454)** and **G.RAMASUJITH (Reg No: 38110447)** hereby declare that the Project Report entitled “**CNN BASED RAIL SURFACE CRACK DETECTION USING DIFFERENT MACHINE LEARNING TECHNIQUES**” done by us under the guidance of **DR. P. JEYANTHI** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in 2018-2022.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.Sasikala M.E.,Ph.D., Dean**, School of Computing **Dr.S.Vigneshwari M.E., Ph.D.** and **Dr.L.Lakshmanan M.E., Ph.D.** , Heads of the Department of Computer Science and Engineering for providing us necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and a deep sense of gratitude to my Project Guide **DR. P. JEYANTHI** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

The proposed system utilizes CNN to detect the faults in the railway tracks with the help of images. Their characteristics are obtained to extract the default railway track. This model helps to reduce the manual inspection work. While performing the task, it lowers the time and cost utilization and increases the safety of passengers. To train and validate the CNN algorithm, the ground truth databases on images of masks are utilized, this proposed model uses different machine learning techniques and classifications like Random Forest Classifier, Gradient Classification, etc .To this end, this CNN maintains the both space and time coherence of the defects in the tracks, and the final output of the false prediction effectively.

TABLE OF CONTENTS

Chapter No.	TITLE	Page No.
1	INTRODUCTION	8
	1.1. OVERVIEW	8
	• . MACHINE LEARNING	9
	• .OBJECTIVE •	9
2	LITERATURE SURVEY	10
	2.1 RELATED WORK	10
3	METHODOLOGY	15
	3.1 EXISTING SYSTEM	15
	3.1.1 DISADVANTAGES EXISTING SYSTEM	15
	3.2 PROPOSED SYSTEM	15
	3.2.1 ADVATAGES OF PROPOSED SYSTEM	16
	3.3 ALGORITHMS USED	16

[illegible]

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The detection of defects or cracks in rail track plays an important role in railway management, which prevents train accidents in both summer and rainy seasons. During summer, the cracks are formed on the track which slips the train wheel. In rainy environment, the rail tracks are affected by corrosion which also produced cracks on it. In present method, the cracks or defects are detected Echo image display device or semiconduction magnetism sensor devices which consumes more time. The proposed method enhances the track image using adaptive histogram equalization technique and further feature are extracted from the enhanced rail track image. In this proposes system we demonstrate that CNN can be used to localize cracks on railway track surfaces and obtain their corresponding masks to aid extract other properties that are useful for inspection. Such a tool can help mitigate the drawbacks of manual

inspection by automating crack detection, lowering time consumption in executing this task, reducing costs and increasing the safety of the personnel. To train CNN for crack detection we built a groundtruth database of masks on images from a subset of a standard crack dataset. To this end, a CNN is proposed to detect crack patches in images, while the proposed data fusion scheme maintains the spatiotemporal coherence of cracks in images, and the decision making discards false positives effectively.

1.2 MACHINE LEARNING

Machine learning could be a subfield of computer science (AI). The goal of machine learning typically is to know the structure information of knowledge of information and match that data into models which will be understood and used by folks. Although machine learning could be a field inside technology, it differs from ancient process approaches.

In ancient computing, algorithms are sets of expressly programmed directions employed by computers to calculate or downside solve. Machine learning algorithms instead give computers to coach on knowledge inputs and use applied math analysis so as to output values that fall inside a particular vary. thanks to this, machine learning facilitates computers in building models from sample knowledge so as to modify decision-making processes supported knowledge inputs.

1.3 OBJECTIVE

In modern technology , the main objective of this project is to detect whether there are any defects or cracks in the image of the Railway Tracks using Convolutional Neural Networks(CNN). Here we use CNN to localize cracks on Railway surfaces to obtain corresponding masks to aid extract other properties that are useful for inspection.

CHAPTER 2

LITERATURE SURVEY

2.1 RELATED WORK

[2.1] Inspection System for Rail Surfaces Using Differential Images

AUTHORS: F. J. de la Calle, Daniel F. Garcia , Ruben Usamentiaga

The acquisition system uses the technique of spectral image differentiation. This technique is used to detect volumetric defects on the surface such as protrusions or scratches. This technique works as follows. Two light sources are placed before and after a line scan camera. These light sources may be of different colors; in this case red and blue light sources are used. Flat surfaces will reflect the same amount of light from both sources, independently of their texture. Surfaces with volumetric defects will reflect different amounts of light from each source due to the shadows that defects create. Using a color camera it is possible to divide the image in two channels, red and blue, and compare the light reflected from each source. Assuming that both light sources generate the

same amount of light over the surface and that the rail has the same reflective properties all over its surface, the difference between the red channel and the blue channel will be zero on flat surfaces and non zero on defects. The proposed system is an open solution for the inspection of rail surface, easy to implement at an affordable cost, which can be systematically configured by the quality engineers of the manufacturing company.

[2.2] A Hierarchical Extractor-Based Visual Rail Surface Inspection System

AUTHORS: JinruiGan ,Qingyong Li, Member, IEEE, Jianzhu Wang, and Haomin Yu

This paper presents an automatic visual inspection system to address these difficulties. Rail surface images are first obtained by the system, and then fed into the proposed hierarchical inspection framework containing coarse extractor and fine extractor. Specifically, the coarse extractor handles this defect inspection problem by exploring characteristics of rail background instead of defect itself, and focuses on finding the background modes. The fine extractor, which integrates the longitudinal context information and transversal prior information, is presented to largely reduce the impacts of other irregulars. The proposed method is evaluated on labeled Type-I and Type-II rail surface discrete defects data sets and a practical rail line. The experimental results demonstrate that our method outperforms the state-of-the-art works. In this part, fine extractor is utilized to determine whether point from outliers obtained above is true defect or other noise point. In order to robustly suppress the influence of noise points, the background modes are fully used to define a longitudinal context information term; The centroids of true defects in the

datasets are surveyed to define a transversal prior information term. Both the two information terms are integrated together into a saliency function, from which we can easily filter these un-required noise points. Mean shift is a nonparametric feature-space analysis technique to locate the modes of the density where the gradient is zero. It is an iterative method, which is described in Algorithm 1. We experimentally set $h = 3$ in the following experiments. Note that the computation of the mean shift vector is still based on the entire sample set S . Consequently, all the information captured in the feature space is well exploited, and the quality of the gradient density estimate is not diminished by the use of representative points.

[2.3] Analysis of Rail Erosion Damage During Electromagnetic Launch

AUTHORS: Bin Cao, Wei Guo, Xia Ge, Xiaochao Sun, Mingtao Li, ZizhouSu, Wei Fan, and Juxiang Li

This paper develops a formula to calculate the temperature rise of rail surface, proposes the method of using temperature rise of rail surface for representing rail erosion damage, and the mathematical formula indicates the relevance between the temperature rise of rail surface and current, linear current density, armature moving speed, rail material, state of armature-rail contact, and skin effect. Dynamic test data required for calculation of rail current, muzzle voltage, armature bore displacement, and velocity are obtained through launching tests, and are used to calculate the surface temperature rise of rails of different materials during the launch of EMRGs. The calculation produces the temperature rise of rails at any positions along the length direction. The results are compared with the photographs of rail surface after the launch, and the analysis verifies the feasibility of representing rail erosion damage with temperature rise of rail surface. On a self-made small electromagnetic launcher

platform, rails made of brass alloy, copper alloy, and aluminum alloy are used in actual launching tests. Rails of each material go through three launches at the same energy level. The experiment data related to rail current, muzzle voltage, and armature velocity and the mechanical and electrical parameters of the launcher are obtained and substituted into the theoretical model to calculate the transient temperature rise of rails made of brass, copper, and aluminum during launching process. The calculated maximum temperature rise of the second launch occurs at 345 mm, reaching 500.8 °C. It is the same as what the first launch presents. Another significant temperature rise occurs at 838 - 1018 mm. The photograph shows significantly more serious erosion damage at the range than at other positions. The calculated maximum temperature rise of the third launch occurs at 475 mm, reaching 537.2 °C. Different from the first two launches, the starting point where metal liquefaction layer is generated, is less obvious. It is found through the analysis that there is an effective armature-rail contact when the armature is started, and the temperature rise calculated by the model is low, rendering the starting point, where metal liquefaction layer is generated less obvious. The calculation result shows the maximum temperature rise at the range between 433 and 553 mm, whereas the corresponding range in the photograph has the most serious erosion; the temperature rise is intermediate at 1158 mm, whereas the corresponding range in the photograph shows relatively minor erosion; and the temperature rise is small at 1392 mm, whereas the corresponding range in the photograph shows the lightest erosion damage.

[2.4] : Impact of Corroded Copper Rails on the Performance of a Miniature Electromagnetic Launcher

AUTHORS: C. G. Gnegy-Davidson, D. A. Wetz, and D. Wong

In this paper, The U.S. Navy is getting much closer to fielding shipboard railguns. As they get closer, there is increasing concern with how corrosion of the rail surface will affect launcher performance and lifetime, with it intuitively hypothesized that corrosion will negatively affect both. It is anticipated that significant surface-oxide formation on the rail surface will increase the rail/armature contact resistance, and surface pitting/deposition will further affect the rail/armature contact area. There are several negative effects that could arise from surface corrosion on the rails. One includes an increased probability of armature transition occurring that can significantly decrease launcher efficiency and muzzle velocity, and enhance the rate of rail erosion. In the experiments discussed here, the University of Texas at Arlington has performed two sets of experiments aimed at better understanding how corrosion impacts the performance of a 1-m long railgun with muzzle energies of roughly 1.8 kJ (14 g to roughly 450 m/s). In the first experiment, virgin C11000 rails were shot on five times, and after the shots, the rails were segmented and characterized using an optical microscope and X-Ray diffraction. In the second set of experiments, bare C11000 rails were corroded, using a custom salt fog chamber for 96 h, before being shot on five times and then surface characterized in the same manner. The launcher has a peak current of roughly 100 kA, and launch packages fabricated from 6061-T6 aluminum are used. The rail claddings studied have dimensions of 1 in $\times$ 39.4 in $\times$ 1/8 in. The railgun performance was evaluated using the standard measurements of armature current and muzzle voltage. The electrical data presented here is used as a performance metric to evaluate launcher performance across successive shots and each of the two experimental series. The two measurements made include launcher current and muzzle voltage, respectively. A decrease in peak current is observed when arc transition occurs. This stems from the increased resistivity that is present when this type of armature/rail contact occurs. Spikes in the muzzle voltage measurement indicate a loss in armature/rail contact and the presence of arcing,

known in the community as armature “transition.” Fig. 6 shows the electrical data collected during the first shot taken on the bare C11000 rails used in the “baseline” series and the first shot performed on the “corroded” rails. As expected, the first shot taken on bare copper rails has the best overall performance with the higher peak current of 104 kA and a muzzle voltage trace without any transition contact observed. Corrosion on the rails significantly decreased the performance of the launcher when compared with the shot performed on bare copper rails. Electrical data collected during the third shot of each respective series of five shots are shown in Fig. 7. It should be noticed that there is no real change in the performance of the bare copper series, as expected. There is significant improvement of the “corroded” rail, where both the current and muzzle voltage traces are closely on top of those recorded from the bare copper experiment. The arcing has decreased to the point where there is still significant energy left in the launcher when the projectile exits. The improved performance is attributed to the armature, “cleaning” the rails by scraping some of the oxidation off the surface as it slides over it, heating the oxidation products significantly and the deposition of new aluminum on the corroded surface.

CHAPTER 3

METHODOLOGY

3.1 EXISTING SYSTEM

- The proposed system utilizes CNN to detect the faults in the railway tracks with the help of images. Their characteristics are obtained to extract the default railway track. This model helps to reduce the manual inspection work. While performing the task, it lowers the time and cost utilization and increases the safety of passengers.
- To train and validate the CNN algorithm, the ground truth databases on images of masks are utilized. To this end, this CNN maintains the both space and time coherence of the defects in the tracks, and the final output of the false prediction effectively. In the existing system, the image processing and deep neural network-based CNN model is used for detecting the faulty images in the railway crack detection. This inspection process is performed by collecting the images of railway tracks by Otsu segmentation model mainly used in railway surface detection.

- When variance is calculated by rail surface image the signal is obtained it will determine whether there is any fault in the particular image of the railway track surface. When the dataset is loaded the image processing will take place by reading, resizing and delete the all noises in the images, divide into defined categories and finally predict whether there is any default or not in the image of railway track.
- Actually, a promising defect inspector (i.e., the definition of distance measurement shown in our reformulation) is assumed to be able to highlight the defects, so a key process is to choose a proper background representation.
- A simple way is to generate a global inspector always using all pixels detected as background pixels. As a result, the defect pixels contaminate the background representation. The aim of the defect inspector is to select adequate and representative background pixels from the rail surface image and define a proper measurement for the background while limiting the effect of defects.

3.1.1 DISADVANTAGES EXISTING SYSTEM

- Time Consuming
- Labor Intensive
- Low Accuracy
- Error Prone

3.2 PROPOSED SYSTEM

In the proposed, an CNN based image processing based method for detecting faults on railway track surfaces is proposed. The rail surface is detected by taking the rail surface images from the railway. Otsu segmentation method is

used in rail surface detection process. The signal is obtained by calculating the variance of the obtained rail surface image. By taking the signal obtained, it is determined whether there is a fault on the rail surface. The type of fault is determined by extracting features from variance signals belonging to faulty images.

In the proposed system, first the histogram equalization is used to reduce the effect of uneven illumination on the detection of cracks. Secondly, the improved Canny edge detection algorithm is used to draw a continuous crack profile with adaptive dual thresholds. Then, an algorithm based on the contour of the fracture shape is used to extract the fracture characteristics and achieve accurate crack location. Finally, a criterion for determining the level of cracking faults based on the width of cracks is proposed. The algorithm can effectively improve the accuracy of track plate crack detection under complex environment. It was calculated using the ratio of the ‘true cracks’ detected to the total number of cracks identified. On the other hand, Recall is a measure of how much of the actual cracks were detected. It was calculated using the ratio of the ‘true cracks’ detected to the total number of actual cracks.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM

- Works well with high dimensionality
- Works well with large dataset
- Effectively improve prediction accuracy.
- Reduces the consumption of hardware resources
- Computationally Efficient

3.3 ALGORITHMS USED

3.3.1 RANDOM FOREST ALGORITHM

Random forest algorithm can use both for classification and the regression kind of problems. In this you are going to learn, how the **random forest algorithm** works in machine learning for the classification task.

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning**, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

A random forest algorithm consists of many decision trees. The ‘forest’ generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The below diagram explains the working of the Random Forest algorithm:

Random Forest Steps

- Randomly select “k” features from total “m” features.

- Where $k \ll m$
- Among the “k” features, calculate the node “d” using the best split point.
- Split the node into daughter nodes using the best split.
- Repeat 1 to 3 steps until “l” number of nodes has been reached.
- Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees.

The beginning of random forest algorithm starts with randomly selecting “k” features out of total “m” features. In the image, you can observe that we are randomly taking features and observations.

3.3.2 KNN ALGORITHMS

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.

- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

3.4 HARDWARE REQUIREMENTS

System : Pentium i3 Processor

Hard Disk : 500 GB.

Monitor : 15'' LED

Input Devices : Keyboard, Mouse

Ram : 2 GB

3.5 SOFTWARE REQUIREMENTS

Operating system : Windows 10

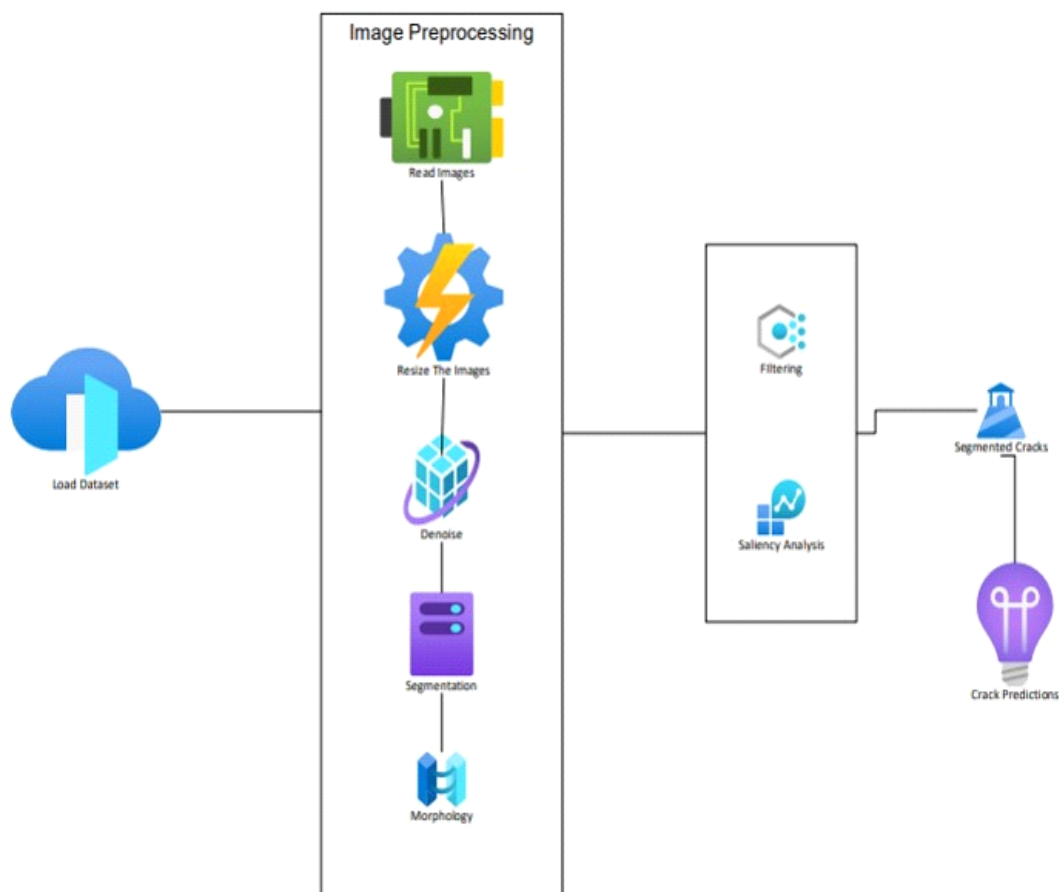
Coding Language : Python

3.6 DIAGRAM

DATA FLOW DIAGRAM

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

- The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

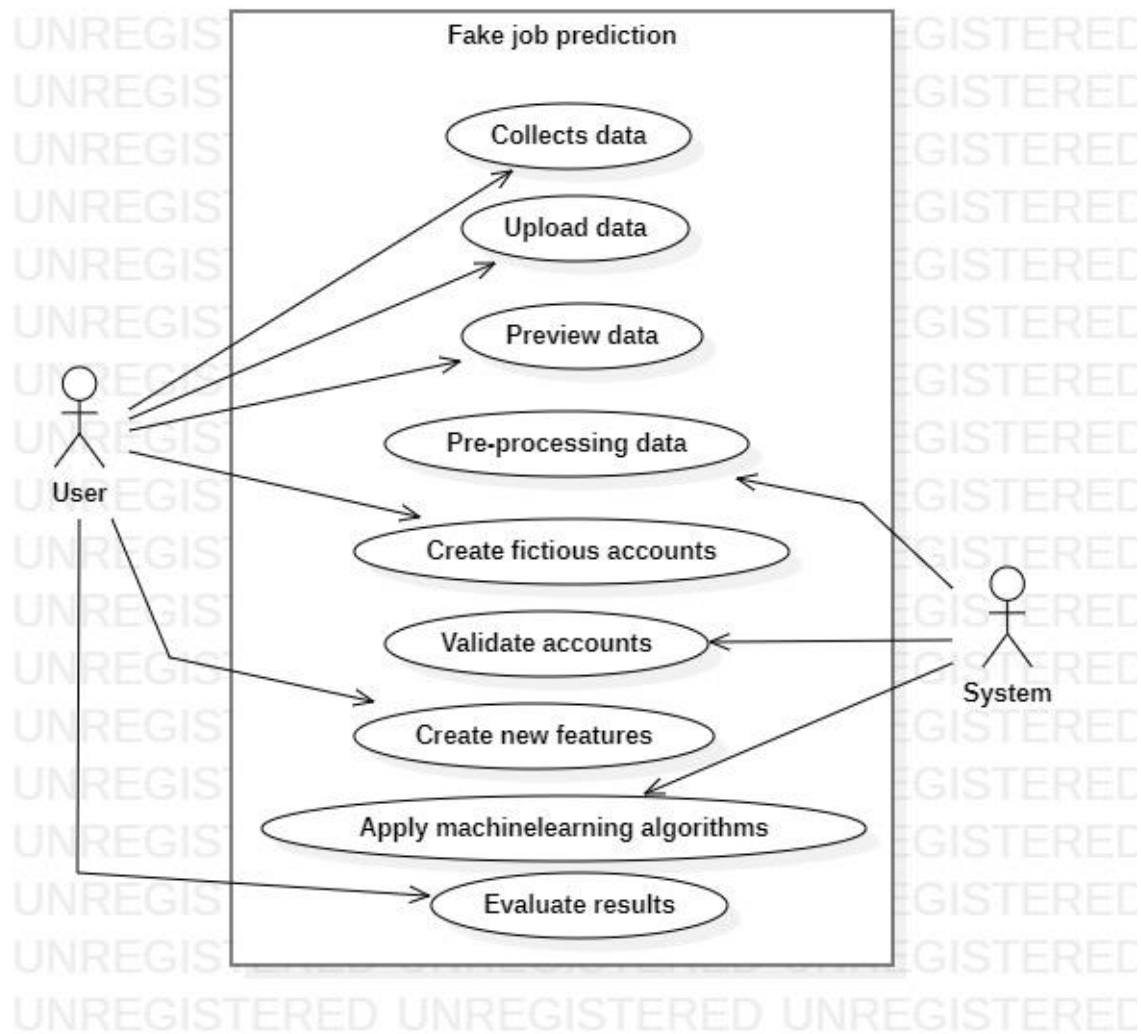
The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.

- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components.
- Integrate best practices.

USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



ACTIVITY DIAGRAM:

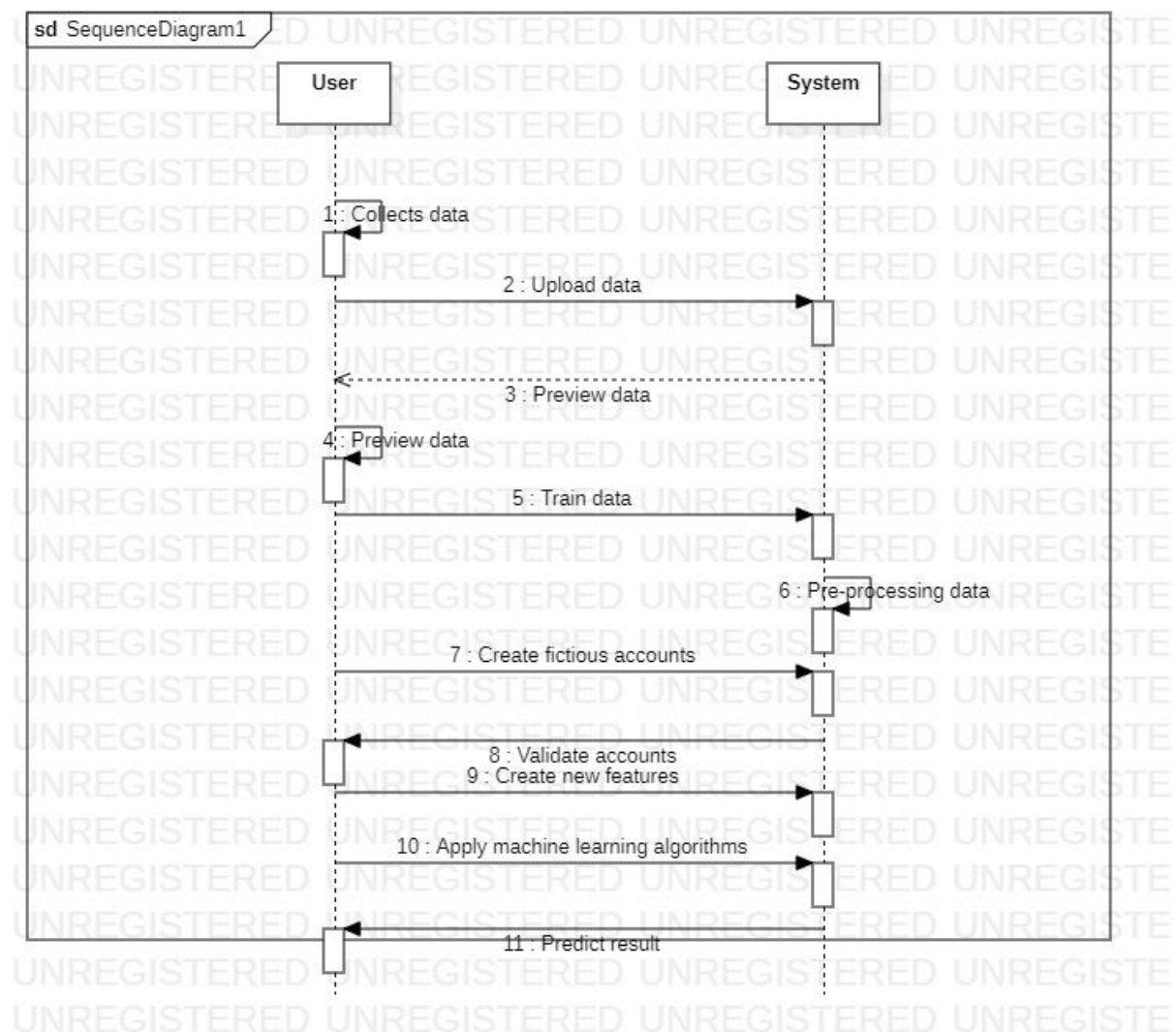
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the

Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a

system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

3.7 MODULES

- Data Collection
- Pre-Processing
- Train and Test
- Machine Learning Technique
- Detection of Fake Job post

MODULE DESCRIPTION

3.7.1 Data Collection

In this step data set with are considered as input has many fields in that except last one all are called as features and last cell is called as labels which are in to 0 1 format.

3.7.2 Pre-Processing

We convert data in to scalar format and then create new features which are passed to algorithm and features are saved in x and labels in y.

3.7.3 Train and Test

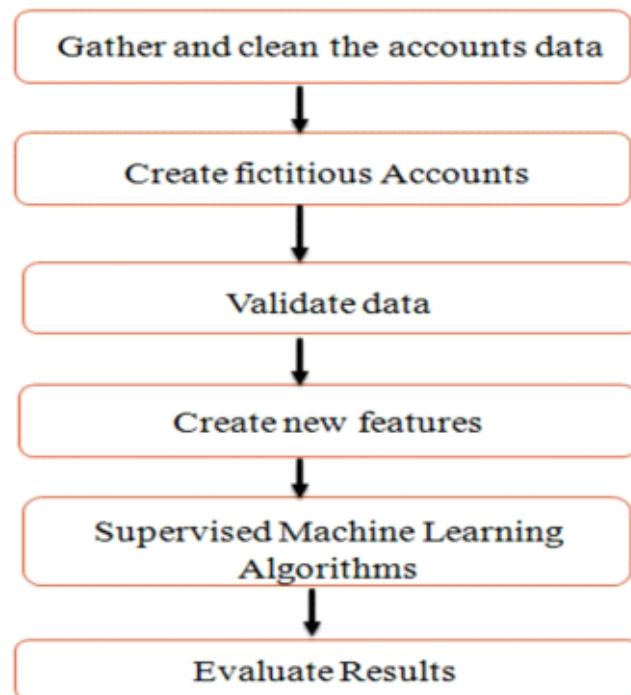
we will split the dataset into training dataset and test dataset. We will use 70% of our data to train and the rest 30% to test. To do this, we will create a split parameter which will divide the data frame in a 70-30 ratio.

3.7.4 Machine Learning Technique

After splitting the dataset into training and test dataset, we will instantiate Random Forest Classifier and KNN classifier fit the train data by using ‘fit’ function. Then we will store as model.

3.8 SYSTEM ARCHITECTURE

Describing the overall features of the software is concerned with defining the requirements and establishing the high level of the system. During architectural design, the various web pages and their interconnections are identified and designed. The major software components are identified and decomposed into processing modules and conceptual data structures and the interconnections among the modules are identified. The following modules are identified in the proposed system.



The above architecture describes the work structure of the system.

Proposed system is equipped with various Machine Learning tasks and the architecture followed is as shown below. The proposed system collects the dataset which are preprocessed by providing a framework of algorithms using which we can detect fake job post in Facebook by comparing the accuracy of three machine learning algorithms and the algorithm with very high efficiency is found for the given dataset. The different ways in which an algorithm can model a problem is based on its interaction with the experience or environment for the model preparation process that helps in choosing the most appropriate algorithm for the given input data in order to get the best result.

3.8.1 Problem Statement:

In modern technology , the main objective of this project is to detet whether there are any defects or cracks in the image of the Railway Tracks using

Convolutional Neural Networks(CNN). Here we use CNN to localiecrackson Railway surfaces to obtain corresponding masks t aid extract other properties that arre useful for inspection.

INPUT DESIGN AND OUTPUT DESIGN

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in

the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

3.9 LANGUAGE USED

Python:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

What is Python?

Python is a popular programming language. It was created in 1991 by Guido van Rossum.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

- Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- Python was designed to for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Python Install

Many PCs and Macs will have python already installed.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

```
C:\Users\Your Name>python --version
```

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

```
python --version
```

If you find that you do not have python installed on your computer, then you can download it for free from the following website: <https://www.python.org/>

Python Quickstart

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

```
C:\Users\Your Name>python helloworld.py
```

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

```
helloworld.py
```

```
print("Hello, World!")
```

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

```
C:\Users\Your Name>python helloworld.py
```

The output should read:

```
Hello, World!
```

Congratulations, you have written and executed your first Python program.

The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

```
C:\Users\Your Name>python
```

From there you can write any python, including our hello world example from earlier in the tutorial:

```
C:\Users\Your Name>python
```

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>> print("Hello, World!")
```

Which will write "Hello, World!" in the command line:

```
C:\Users\Your Name>python
```

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
```

```
Type "help", "copyright", "credits" or "license" for more information.  
>>> print("Hello, World!")  
Hello, World!
```

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

```
exit()
```

Execute Python Syntax

As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

```
>>>print("Hello,                                     World!")  
Hello, World!
```

Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:

```
C:\Users\Your Name>python myfile.py
```

Python Indentations

Where in other programming languages the indentation in code is for readability only, in Python the indentation is very important.

Python uses indentation to indicate a block of code.

Example

```
if 5 > 2:  
    print("Five is greater than two!")
```

Python will give you an error if you skip the indentation:

Example

```
if 5 > 2:  
print("Five is greater than two!")
```

Comments

Python has commenting capability for the purpose of in-code documentation.

Comments start with a #, and Python will render the rest of the line as a comment:

Example

Comments in Python:

```
#This is a comment.  
print("Hello, World!")
```

Docstrings

Python also has extended documentation capability, called docstrings.

Docstrings can be one line, or multiline.

Python uses triple quotes at the beginning and end of the docstring:

Example

Docstrings are also comments:

```
"""This is a  
multiline docstring."""  
print("Hello, World!")
```

CHAPTER 4

SYSTEM STUDY

4.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the

methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing,

that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.1 Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format

- No duplicate entries should be allowed
- All links should take the user to the correct page.

6.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.3 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER-5

CONCLUSION

5.1 CONCLUSION

In this reserach we adapt the state of the art detection model Mask R-CNN to automate crack detection on concrete surfaces. We train this model using our own groundtruth dataset which was built on a subset of images from a recent benchmark annotated image set aimed for training and validation of artificial intelligence based crack detection algorithms for concrete. In order to adapt our framework to scenarios with more varying surfaces and lighting conditions, in the future we aim to retrain the network on a larger and more varied dataset.

REFERENCES

[1] H. Oliveira and P. L. Correia, “Automatic road crack detection and characterization,” IEEE Transactions on Intelligent Transportation Systems, vol. 14, no. 1, pp. 155–168, March 2013,

doi: <https://doi.org/10.1109/TITS.2012.2208630>

[2] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, “A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure,” vol. 29, no. 2, 2015, pp. 196 – 210, infrastructure ComputerVision.[Online].

Available:

<http://www.sciencedirect.com/science/article/pii/S1474034615000208>

[3] K. Kang et al., “T-CNN: Tubelets with convolutional neural networks for object detection from videos,” IEEE Trans. Circuits Syst. Video Technol., to be published, 2016.

[4] O. Russakovsky et al., "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[5] D. Soukup and R. Huber-Mork, "Convolutional neural networks for steel surface defect detection from photometric stereo images," in *Proc. Int. Symp. Visual Comput.*, Dec. 2014, pp. 668–677.

[6] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2016, pp. 3708–3712.

[7] Y.-J. Cha, W. Choi, and O. Buyukozturk, "Deep learning-based crack damage detection using convolutional neural networks," *Comput.-Aided Civil Infrastructure Eng.*, vol. 32, no. 5, pp. 361–378, Mar. 2017.

[8] S. J. Schmugge et al., "Detection of cracks in nuclear power plant using spatialtemporal grouping of local patches," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2016, pp. 1–7.

[9] Sireesha R, Ajay Kumar B, Mallikarjunaiah G and Bharath Kumar B, "Broken Rail Detection System using RF Technology", in "SSRG International Journal of Electronics and Communication Engineering", volume 2 issue 4- April 2015

[10] Prof. P.Navaraja, "CRACK DETECTION SYSTEM FOR RAILWAY TRACK BY USING ULTRASONIC AND PIR SENSOR", in "INTERNATIONAL JOURNAL OF ADVANCED INFORMATION AND COMMUNICATION TECHNOLOGY", Volume -1, Issue-1, May 2014

APPENDICES

- PLAGIARISM REPORT

A COMPARATIVE STUDY ON CNN B..

ORIGINALITY REPORT

9%

SIMILARITY INDEX

9%

INTERNET SOURCES

6%

PUBLICATIONS

5%

STUDENT PAPERS

PRIMARY SOURCES

1

"Table of Contents", IEEE Transactions on Computational Social Systems, 2015

Publication

1%

2

www.jpinfotech.org

Internet Source

1%

3

Shivangi Gheewala, Rakesh Patel. "Machine Learning Based Twitter Spam Account Detection: A Review", 2018 Second International Conference on Computing Methodologies and Communication (ICCMC), 2018

Publication

1%

4

researchr.org

Internet Source

1%

5

www.ijrte.org

Internet Source

1%

6

www.pantechelearning.com

Internet Source

1%

7

www.ijert.org

Internet Source

1%

8	www.iosrjournals.org Internet Source	1 %
9	Submitted to Victoria University Student Paper	1 %
10	bionyt.s807.sureserver.com Internet Source	<1 %
11	jpinfotech.org Internet Source	<1 %
12	cps-vo.org Internet Source	<1 %
13	ijettjournal.org Internet Source	<1 %
14	computer.ieeeprojects.org Internet Source	<1 %
15	Faiza Masood, Ghana Ammad, Ahmad Almogren, Assad Abbas, Hasan Ali Khattak, Ikram Ud Din, Mohsen Guizani, Mansour Zuair.	<1 %

SOURCE CODE and SCREENSHOTS

Home Page - Select or create a notebook | Railway-Track-Fault-Detection-NN | Microsoft Edge | localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb | YouTube | Gmail

jupyter Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (unsaved changes) | Trusted | Python 3

File Edit View Insert Cell Kernel Widgets Help

In [1]: `import os`

In [2]: `%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.image as mpimg`

In [3]: `import plotly.express as px`

In [4]: `import pandas as pd`

In [5]: `import os`

In [6]: `from tensorflow.keras import layers`

C:\Users\ramas\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:516: FutureWarning: Passing (type, 1) or 'type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
_np_qint8 = np.dtype [("qint8", np.int8, 1)]
C:\Users\ramas\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:517: FutureWarning: Passing (type, 1) or 'type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
_np_quint8 = np.dtype [("quint8", np.uint8, 1)]
C:\Users\ramas\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:518: FutureWarning: Passing (type, 1) or 'type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
_np_qint16 = np.dtype [("qint16", np.int16, 1)]
C:\Users\ramas\Anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:519: FutureWarning: Passing (type, 1) or 'type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
_np_resource = np.dtype [("resource", np.ubyte, 1)]

In [7]: `from tensorflow.keras import Model`

33°C Partly sunny | ENG US | 09:34 08-05-2022

Microsoft Edge | localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb | Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (autosaved)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3

```
In [7]: from tensorflow.keras import Model

In [8]: from tensorflow.keras.applications.inception_v3 import InceptionV3

In [9]: import tensorflow as tf

In [10]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

In [11]: from tensorflow.keras.optimizers import RMSprop

In [12]: import cv2

In [13]: import numpy as np

In [14]: import pandas as pd

In [15]: import random

In [16]: from tensorflow.keras.preprocessing.image import img_to_array, load_img

In [17]: train_dir = os.path.join('dataset/Train')
validation_dir = os.path.join('dataset/Validation')

In [18]: train_defective_dir = os.path.join('dataset/Train/Defective')
```

33°C Partly sunny | ENG US | 09:37 08-05-2022

Microsoft Edge | localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb | Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3

```
In [18]: train_defective_dir = os.path.join('dataset/Train/Defective')
train_nondefective_dir = os.path.join('dataset/Train/Non defective')

In [19]: validation_defective_dir = os.path.join('dataset/Validation/Defective')
validation_nondefective_dir = os.path.join('dataset/Validation/Non defective')

In [20]: train_defective_filenames = os.listdir(train_defective_dir)
train_nondefective_filenames = os.listdir(train_nondefective_dir)

In [21]: print(train_defective_filenames[:20])
print(train_nondefective_filenames[:20])

['IMG_20201114_100206.jpg', 'IMG_20201114_100226.jpg', 'IMG_20201114_100946.jpg', 'IMG_20201114_100947.jpg', 'IMG_20201114_101121.jpg', 'IMG_20201114_101130.jpg', 'IMG_20201114_101134.jpg', 'IMG_20201114_101143.jpg', 'IMG_20201114_101148.jpg', 'IMG_20201114_101157.jpg', 'IMG_20201114_101539.jpg', 'IMG_20201114_101541.jpg', 'IMG_20201114_101612.jpg', 'IMG_20201114_101628.jpg', 'IMG_20201114_101630.jpg', 'IMG_20201114_101637.jpg', 'IMG_20201114_101658.jpg', 'IMG_20201114_101820.jpg', 'IMG_20201114_101823.jpg', 'IMG_20201114_101930.jpg']
['IMG_20201114_100032.jpg', 'IMG_20201114_100044.jpg', 'IMG_20201114_100050.jpg', 'IMG_20201114_100052.jpg', 'IMG_20201114_100101.jpg', 'IMG_20201114_100107.jpg', 'IMG_20201114_100114.jpg', 'IMG_20201114_100130.jpg', 'IMG_20201114_100139.jpg', 'IMG_20201114_100140.jpg', 'IMG_20201114_100215.jpg', 'IMG_20201114_100230.jpg', 'IMG_20201114_100242.jpg', 'IMG_20201114_100245.jpg', 'IMG_20201114_100248.jpg', 'IMG_20201114_100259.jpg', 'IMG_20201114_100301.jpg', 'IMG_20201114_100310.jpg', 'IMG_20201114_100315.jpg', 'IMG_20201114_100324.jpg']

In [22]: print('total training defective images :', len(os.listdir(train_defective_dir)))
print('total training non-defective images :', len(os.listdir(train_nondefective_dir)))

total training defective images : 54
```

33°C Partly sunny | ENG US | 09:37 08-05-2022

Home Page - Select or create a notebook | Railway-Track-Fault-Detection-NN | Microsoft Edge | localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb

Jupyter Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
In [22]: print('total training defective images :', len(os.listdir(train_defective_dir)))
print('total training non-defective images :', len(os.listdir(train_nondefective_dir)))

total training defective images : 54
total training non-defective images : 136

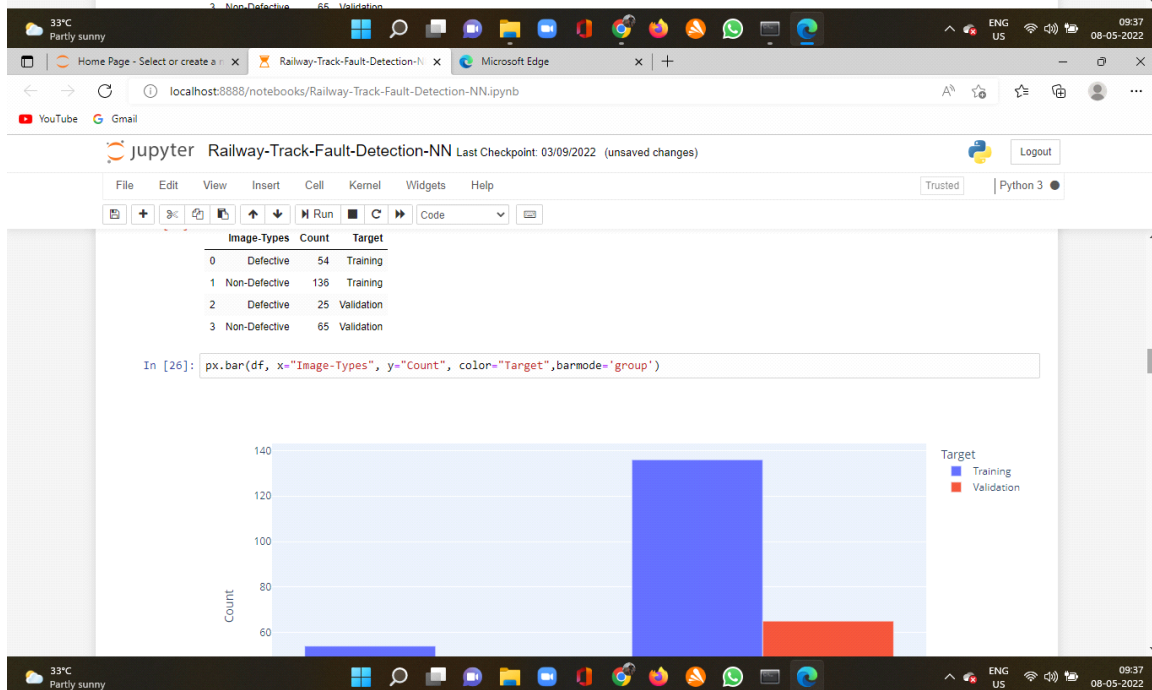
In [23]: print('total validation defective images :', len(os.listdir( validation_defective_dir ) ))
print('total validation non-defective images :', len(os.listdir( validation_nondefective_dir ) ))

total validation defective images : 25
total validation non-defective images : 65

In [24]: df = pd.DataFrame([["Defective",len(os.listdir(train_defective_dir)), "Training"],
                           ["Non-Defective",len(os.listdir(train_nondefective_dir)), "Training"],
                           ["Defective",len(os.listdir(validation_defective_dir)), "Validation"],
                           ["Non-Defective",len(os.listdir(validation_nondefective_dir)), "Validation"]], columns=["Image-Types", "Count"]

In [25]: df
Out[25]:
```

	Image-Types	Count	Target
0	Defective	54	Training
1	Non-Defective	136	Training
2	Defective	25	Validation
3	Non-Defective	65	Validation



Microsoft Edge | Railway-Track-Fault-Detection-NN | Last Checkpoint: 03/09/2022 (unsaved changes) | Python 3

```
In [27]: nrows = 4
ncols = 4

In [28]: pic_index = 0

In [29]: pic_index += 8
next_defective_pix = [os.path.join(train_defective_dir, fname)
                     for fname in train_defective_fnames[pic_index-8:pic_index]]
next_nondefective_pix = [os.path.join(train_nondefective_dir, fname)
                        for fname in train_nondefective_fnames[pic_index-8:pic_index]]

In [30]: fig = plt.gcf()
fig.set_size_inches(ncols * 4, nrows * 4)
for i, img_path in enumerate(next_defective_pix+next_nondefective_pix):
    sp = plt.subplot(nrows, ncols, i + 1)
    sp.axis('off')

    img = mpimg.imread(img_path)
    plt.imshow(img)

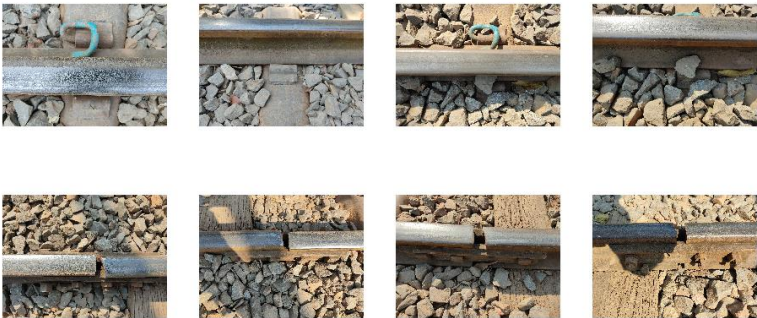
plt.show()
```

33°C Partly sunny | 09:37 | 08-05-2022

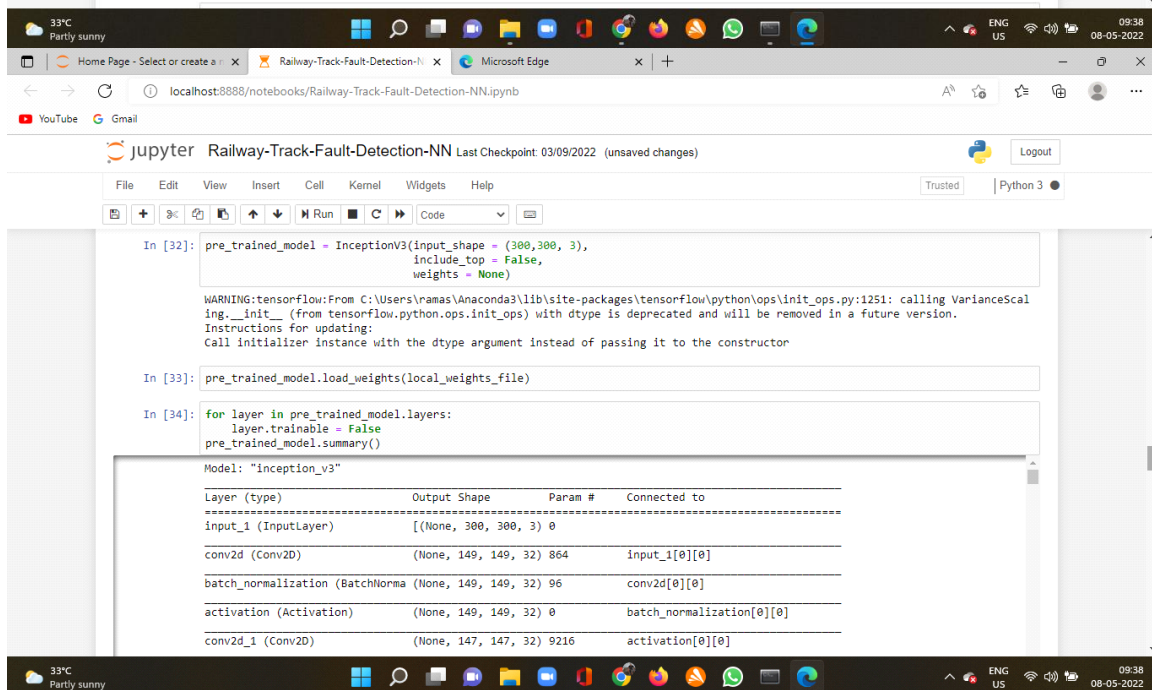
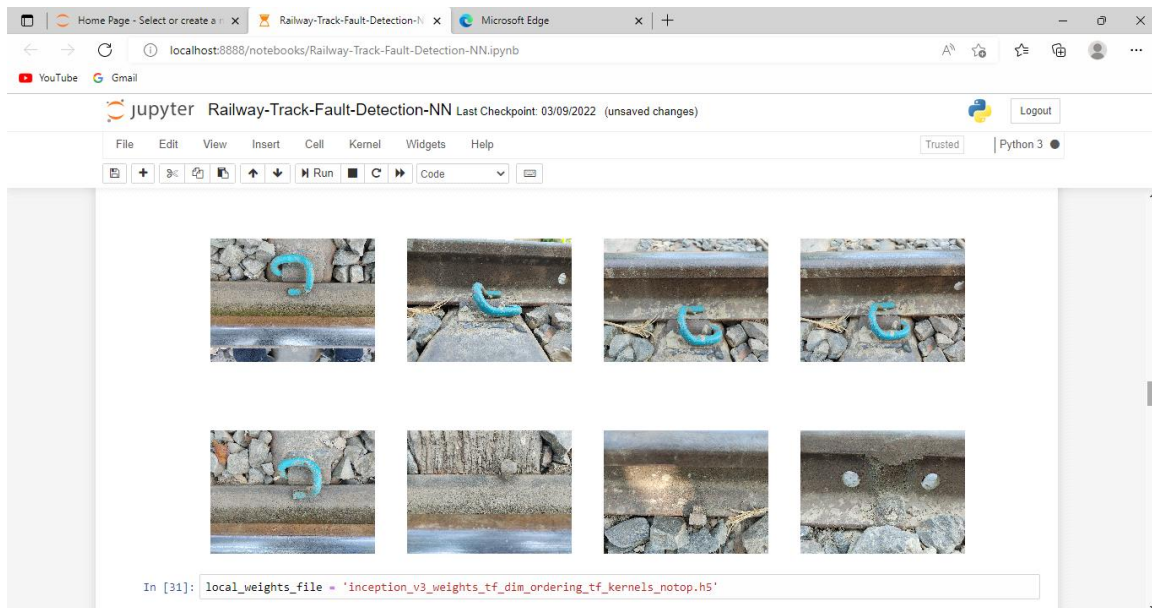
Microsoft Edge | Railway-Track-Fault-Detection-NN | Last Checkpoint: 03/09/2022 (unsaved changes) | Python 3

```
img = mpimg.imread(img_path)
plt.imshow(img)

plt.show()
```



33°C Partly sunny | 09:38 | 08-05-2022



Home Page - Select or create a notebook x Railway-Track-Fault-Detection-NN x Microsoft Edge x +

localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb

YouTube Gmail

Jupyter Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Model: "inception_v3"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 300, 300, 3)]	0	
conv2d (Conv2D)	(None, 149, 149, 32)	864	input_1[0][0]
batch_normalization (BatchNormaliza	(None, 149, 149, 32)	96	conv2d[0][0]
activation (Activation)	(None, 149, 149, 32)	0	batch_normalization[0][0]
conv2d_1 (Conv2D)	(None, 147, 147, 32)	9216	activation[0][0]
batch_normalization_1 (BatchNor	(None, 147, 147, 32)	96	conv2d_1[0][0]
activation_1 (Activation)	(None, 147, 147, 32)	0	batch_normalization_1[0][0]
conv2d_2 (Conv2D)	(None, 147, 147, 64)	18432	activation_1[0][0]

```
In [35]: last_layer = pre_trained_model.get_layer('mixed7')

In [36]: print('last layer output shape: ', last_layer.output_shape)

last layer output shape: (None, 17, 17, 768)

In [37]: last_output = last_layer.output
```

33°C Partly sunny

Home Page - Select or create a notebook x Railway-Track-Fault-Detection-NN x Microsoft Edge x +

localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb

YouTube Gmail

Jupyter Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
In [38]: train_datagen = ImageDataGenerator(rescale = 1./255.,
      rotation_range = 40,
      width_shift_range = 0.2,
      height_shift_range = 0.2,
      shear_range = 0.2,
      zoom_range = 0.2,
      horizontal_flip = True)

In [39]: test_datagen = ImageDataGenerator(rescale = 1./255.,
      rotation_range = 40,
      width_shift_range = 0.2,
      height_shift_range = 0.2,
      shear_range = 0.2,
      zoom_range = 0.2,
      horizontal_flip = True)

In [40]: train_generator = train_datagen.flow_from_directory(train_dir,
      batch_size=20,
      class_mode='binary',
      target_size=(300,300))

Found 190 images belonging to 2 classes.

In [41]: validation_generator = test_datagen.flow_from_directory(validation_dir,
      batch_size=20,
      class_mode = 'binary',
      target_size = (300,300))
```

33°C Partly sunny

Microsoft Edge | localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb | Logout

Jupyter Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help | Trusted | Python 3

```
In [*]: history = model.fit(train_generator,
                           validation_data=validation_generator,
                           steps_per_epoch=10,
                           epochs=3,
                           validation_steps=5,
                           verbose=2)

Epoch 1/3
10/10 - loss: 1.1505 - acc: 0.6000 - val_loss: 0.5898 - val_acc: 0.7333
Epoch 2/3
10/10 - loss: 0.5690 - acc: 0.6947 - val_loss: 0.8947 - val_acc: 0.7222
Epoch 3/3

In [*]: acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(len(acc))

plt.plot(epochs, acc)
plt.plot(epochs, val_acc)
plt.title('Training and validation accuracy')
plt.figure()

plt.plot(epochs, loss)
plt.plot(epochs, val_loss)
```

33°C Partly sunny | 09:38 08-05-2022

```
In [*]: img = cv2.imread('dataset/Test/Defective/IMG_20201114_103110.jpg')
plt.imshow(img)
img = cv2.resize(img, (300,300))
img = np.reshape(img, [1,300,300,3])

In [*]: classes = model.predict(img)

In [*]: if classes<0.5:
        print("This Railway track has no fault")
        else:
        print("This Railway track has fault")

In [*]: img = cv2.imread('dataset/Test/Non defective/IMG_20201114_101633.jpg')
plt.imshow(img)
img = cv2.resize(img, (300,300))
img = np.reshape(img, [1,300,300,3])

In [*]: classes = model.predict(img)

In [*]: if classes>0.5:
        print("This Railway track has no fault")
        else:
        print("This Railway track has fault")

In [*]: successive_outputs = [layer.output for layer in model.layers[1:]]
```

33°C Partly sunny | 09:38 08-05-2022

Home Page - Select or create a notebook | Railway-Track-Fault-Detection-NN | Microsoft Edge | localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb | YouTube | Gmail

Jupyter Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (autosaved) | Python 3

```
In [*]: visualization_model = tf.keras.models.Model(inputs = model.input, outputs = successive_outputs)

In [*]: defective_img_files = [os.path.join(train_defective_dir, f) for f in train_defective_fnames]

In [*]: nondefective_img_files = [os.path.join(train_nondefective_dir, f) for f in train_nondefective_fnames]

In [*]: img_path = random.choice(defective_img_files + nondefective_img_files)

In [*]: img = load_img(img_path, target_size=(300,300))

In [*]: x = img_to_array(img)

In [*]: x = x.reshape((1,) + x.shape)

In [*]: x /= 255

In [*]: successive_feature_maps = visualization_model.predict(x)

In [*]: layer_names = [layer.name for layer in model.layers[1:]]

In [*]: for layer_name, feature_map in zip(layer_names, successive_feature_maps):
    if len(feature_map.shape) == 4:
        n_features = feature_map.shape[-1]
        size = feature_map.shape[1]
```

33°C Partly sunny | ENG US | 09:38 08-05-2022

Home Page - Select or create a notebook | Railway-Track-Fault-Detection-NN | Microsoft Edge | localhost:8888/notebooks/Railway-Track-Fault-Detection-NN.ipynb | YouTube | Gmail

Jupyter Railway-Track-Fault-Detection-NN Last Checkpoint: 03/09/2022 (unsaved changes) | Python 3

```
In [*]: x = x.reshape((1,) + x.shape)

In [*]: x /= 255

In [*]: successive_feature_maps = visualization_model.predict(x)

In [*]: layer_names = [layer.name for layer in model.layers[1:]]

In [*]: for layer_name, feature_map in zip(layer_names, successive_feature_maps):
    if len(feature_map.shape) == 4:
        n_features = feature_map.shape[-1]
        size = feature_map.shape[1]
        display_grid = np.zeros((size, size * n_features))
        for i in range(n_features):
            x = feature_map[0, :, :, i]
            x -= x.mean()
            x /= x.std()
            x *= 64
            x += 128
            x = np.clip(x, 0, 255).astype('uint8')
            display_grid[:, i * size : (i + 1) * size] = x
        scale = 20. / n_features
        plt.figure(figsize=(scale * n_features, scale))
        plt.title(layer_name)
        plt.grid(False)
        plt.imshow(display_grid, aspect='auto', cmap='summer')
```

33°C Partly sunny | ENG US | 09:38 08-05-2022

