DEPARTMENT OF CIVIL, ENVIRONMENTAL AND GEOMATIC ENGINEERING

Semester Project Report

Data-driven identificiation and classification of rail surface defectse

Aiyu Liu

Supervised by: Cyprien Hoelzl, Prof. Eleni Chatzi

Sunday 9th February, 2020

Acknowledgements

This semester project would not be possible without the help of my supervisors, Cyprien Hoelzl and professor Eleni Chatzi. I first reached out to professor Chatzi for a semester project opportunity during ETH week

Working alongside Cyprien has been a great experience.

I received all the guidance necessary Whenever I had issues I could always ask and the reply would come promptly provided me with many informative resources very good at explaining concepts very smart and very specialised in this field – huge understanding Can ask any questions, down-to-earth and very helpful. I could not ask for a better supervisor.

Chatzi is very approachable and kind, good at providing feedback at the intermediate sessions.

Contents

Chapter 1

Introduction

1.1 Problem description and motivation

Railway companies need to continuously and sufficiently maintain the train tracks and optimally detect defects in order to have a more punctual and more effective train system. However, the current system is expensive, time consuming and ineffective. That is, maintenance agents need to walk along tracks and check them for defects. For visualisation purposes, there is roughly 5200 km of rails in Switzerland which needs to be inspected by 40 experienced inspectors.

In order to cope with this issue, Swiss Federal Railways (SBB) has specifically built two new special diagnostic vehicles (SDV) designed for defect identification among other purposes. For this, two accelerometers have been installed at the front and back of the vehicle to collect the signal responses from the wheel and the train track

A defect in train tracks can be seen as a discontinuity. As a train passes over this discontinuity, it will result in a perturbation that can be detected by sensors. It is our main assumption that each type of defect will have a specific signature that will allow its identification and classification. This is similar to the idea presented in

By succesfully identifying and classifying the defects, we take one step further towards reducing delays and making trains more punctual and reliable. The first step in this process consists of identification and classification, while the second step consists of future defect prediction.

maybe remove this section

insert picture, mention boogey?

https://blog to-1d-convolutiona neural-networks-in-keras-for-time-sequences-3a7ff801a2cf

1.2 Objective

As the title implies, the objective of this project is to identify and classify rail surface defects.

apply machine learning techniques on the problem

1.3 Defects

Evidently, a defect can be seen as a deviation from the standard train track. For the exact defect type, SBB has self-constructed a database for the individual defect definitions . Here is a few examples:

Generally, a defect is separated into two overarching types: range- and point-defects. I.e. a defect that is detected at a single point versus a defect that is detected at varying

is this a recognized system?

insert pictures

give example

show signal types?

See list of defect types in appendix?

lengths – e.g. .

For this project, we have solely focused on the point defects for analysis, as this simplifies the problem statement. . A point defect is perceived as a sharp signal response, whereas a range-defect is perceived over a greater time period. We thus disregard range-defects such that we do not have to deal with the extra, associated factors.

1.4 Data

The data has been collected and provided by SBB. Using their SDV, SBB has made trips back and forth to different cities in Switzerland in order to collect various data including but not limited to accelerometer data. After getting the data from SBB, it then goes through a processing pipeline (designed by Cyprien), after which the data can be manipulated with `python` dataframes (from `pd.DataFrame`). The accelerometer captures the accelerations at the XYZ-axes (along with the timestamps at each recording), of which we are only concerned with the Y-axis for the vertical pertubations.

make a table of the equipment and sample frequencies

Furhtermore, the locations of the defects have to be retrieved from SBB's database. which were retrieved by Cyprien.

We need to define terminlogy of these: defects, ins joints = in the following we will use defect as an umbrella term for these entities.

1.5 Code

Which data did I work on, put in tables, switches, ins joints and defects

The code is written purely in python. The code can be found on github: https://github.com/Aiyualive/SemesterProject2.0.

Brief explanation of the code?

Chapter 2

Design and Implementation

For the process of defect classification we employed the following pipeline:

In the following, I would like to give an overview of how these was implemented

insert pipeline picture

2.1 Shift of GPS timestamps

which track entities are we actually analysing

The SDV has its GPS sensor installed at a specified location on the vehicle body. However, what we need to achieve is the position (covered distance) at each accelerometer at either sides of the GPS. Since the GPS sensor is sampled at a lower frequency compared to the accelerometers, we first need to get the corresponding positions for each accelerometer sample. This is done by interpolation using the timestamps of the accelerometers and GPS.

Depending on the direction of the vehicle we then subtract/add the offset between the accelerometers and the GPS sensor with regard to the position of these sensors on the vehicle body.

show a few defects and their signals

2.2 Peak windows

appendix for more signals?

Retrieving the signal response around the defect location forms a crucial aspect in the overarching pipeline. The goal of this step is to, around each defect, create a "window" containing accelerometer accelerations of a specified time length – wherein Within the highest acceleration recording around is found in the center. As a result, all of these windows would be uniform in the sense that they are all centered according to the highest recording of a defect. It is then assumed that each window forms the signature of each track entity.

insert drawing of how it is calculated?

Since we are assuming that each track entity is identified by a well-formed peak, we first need to find this peak within a reasonable offset from the defect location, after which we center around that within another reasonable offset.

In the code, this is done by defining two parameters: `find_peak_offset = 1` and `window_offset = 0.5`. I.e. given a defect timestamp, we search for the highest acceleration recording that has occured 1 second after and 1 second before the defect timestamp. Once the peak has been found, we then center it in a 1 second window (0.5 sec on each side).

insert a drawing of how this works?

2.3 Neural network architecture

Using tensorflow, we then feed these windows into our neural network architecture as seen in listing.

insert table

Trained a neural network, although we were only able to achieve max
Create the models and train it
Based on the analysis we

2.4 Visualisation

This step should have been done first

Chapter 3

Evaluation

Here we will present the results and discuss the findings herein.

3.1 Results

Dataset / run #	Synthetic	ClaveVectors	MNIST-bin-digits	MNIST-all-digits
1	14.458	61.355	3212.497	2454.585
2	16.568	62.396	3346.047	2782.271
3	14.362	59.282	2908.446	2618.519
4	13.85	62.193	3043.189	2600.233
5	13.755	61.024	3309.422	2470.139
6	12.218	58.531	3192.178	2823.685
7	12.232	60.187	3287.289	2452.566
8	13.346	59.398	3026.486	2810.943
9	11.433	61.892	3127.583	2528.202
10	11.63	60.925	3018.402	2746.729

Table 3.1: Architecture of model number

3.2 Discussion

I tried to increase the outliers, but this was a hugely naive approach

insert the different defect types and class distributions?

Chapter 4

Conclusion and future work

4.1 Conclusion

4.2 Future work

- Might be interesting to also consider the XZ-axes.
- range defects
- tune the peak finding parameters
- track entity dependent/specific window offsets
- we must not set the findpeakoffset too high
- Questions, what if you want to use multiple features with 1 CNN

4.3 TODO

- get percentage of each class in the validation set
- create an average of the model ie run model for more times
- add speed as a feature
- what does each filter do? what is kernel size?
- very fast speed, overlap between switch and ins, old vs new rail, ax1 arrow 2 arrow 3 arrow 4
- 3D plots?
- change the defect library to use pandas instead?
- visualise what the network is doing using Harry's code
- use speed as a feature also
- be consistent with function naming and variable names: function names with underscore and variable names with camelcase
- save confusion matrix, and metrics (modify the plotting of this)
- citation [1]
- what if we just use evaluate instead of using measuring the accuracy score

Bibliography

[1] Anders Peterson. Java matrix benchmark, 2018. https://github.com/soumith/convnet-benchmarks, visited 2019-15-05.

Appendix A

Results

Dataset	Size	Dimensions	Labels
Synthetic data	2000	9	4 (1 to 4)
ClaveVectors data	10800	16	4 (1 to 4)
MNIST binary digits	12665	784	2 (0 to 1)
MNIST all digits	11000	784	10 (0 to 9)

Table A.1: Dataset formats

Latent Dimension	2
Shape	8×8
Latent Points	64
Dist. Mixtures	Bernoulli
Regularisation α	0.001

RBF Dimension	2
RBF grid	4×4
RBF centres	16
Basis function	Gaussian
Widths $\boldsymbol{\sigma}$	1.0

Table A.2: The parameters for the latent model and the RBF neural network. Distribution mixtures and basis functions cannot be altered as our implementation is specific for the Bernoulli GTM version