DEPARTMENT OF CIVIL, ENVIRONMENTAL AND GEOMATIC ENGINEERING

Data driven identification and classification of rail surface defec

Aiyu Liu

Supervised by: Cyprien Hoelzl, Prof. Eleni Chatzi

Tuesday 4th February, 2020

Acknowledgements

This thesis would not be possible without the help of my supervisors, Cyprien Hoelzl and professor Eleni Chatzi.

Contents

1 Introduction **7**
1.1 Motivation 7
1.2 Objective 7
1.3 Defects 7
1.4 Data 7

2 Design and Implementation **9**
2.1 Peak windows 9
2.2 Neural network architecture 9
2.3 Visualisation 9

3 Evaluation **11**
3.1 Results 11
3.2 Discussion 11

4 Conclusion and future work **13**
4.1 Conclusion 13
4.2 Future work 13
4.3 TODO 14

A Appendix **17**

Chapter 1

Introduction

1.1 Motivation

Railway companies need to continuously and sufficiently maintain the train and the train tracks and opti- mally predict the future defects in order to have a more punctual and more effective train system. However, the current systems are expensive, time consuming and ineffective.

1.2 Objective

1.3 Defects

Defect can be of any type, which defects do we want to focus on

insert pictur

1.4 Data

Data is provided by SBB

Chapter 2

Design and Implementation

First we need to analyse the data,

2.1 Peak windows

To find

2.2 Neural network architecture

Trained a neural network, although we were only able to achieve max Based on the analysis we

2.3 Visualisation

Chapter 3

Evaluation

3.1 Results

3.2 Discussion

Chapter 4

Conclusion and future work

4.1 Conclusion

4.2 Future work

-
-

4.3 TODO

- 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

4.4 Notes

1D convolution tutorial Height = acc length Width = the number of features Output is determined by kernnel size and height of data

Misc:

- `pd.options.display.max_rows = 15`
- `#np.bincount(y.class_label.values)/4 where does 151.5 coem from??`

whats this

```python
def conv(df):
    """
    has to be series
    """
    return np.vstack([v for v in df])

dup_ins = s_features.ins_joints.copy()[['accelerations']]
dup_swi = s_features.switches.copy()[['accelerations']]
dup_def = s_features.defects.copy()[['accelerations']]

dup_ins['accelerations'] = np.sum(conv(dup_ins.accelerations),1)
dup_swi['accelerations'] = np.sum(conv(dup_swi.accelerations),1)
dup_def['accelerations'] = np.sum(conv(dup_def.accelerations),1)

# s_features.ins_joints[['vehicle_speed(m/s)', 'Axle', 'campagin_ID']].duplicated()

idx_ins = dup_ins.accelerations.duplicated()
idx_swi = dup_swi.accelerations.duplicated()
idx_def = dup_def.accelerations.duplicated()
new_ins = s_features.ins_joints[~idx_ins]
new_swi = s_features.switches[~idx_swi]
new_def = s_features.switches[~idx_def]

print("Duplcated samples: ", len(dup_ins) - len(new_ins))
print("Duplcated samples: ", len(dup_swi) - len(new_swi))
print("Duplcated samples: ", len(dup_def) - len(new_def))

# Load weight example
# Could just save entire model and then load entire model
# Could also make this into a function
clf2 = NN(N_FEATURES, N_CLASSES)
clf2.prepare_data(X, y)
clf2.make_model2()
clf2.load_weights('model_01-12-2019_150004.hdf5')
clf2.predict() ### on validation set
clf2.measure_performance(accuracy_score)
```

Test sample

```
ii = pd.DataFrame([
    [np.array([1,2]),2],
    [np.array([1,2]),2],
    [np.array([1,2]),2]])


x = a
[u,I,J] = unique(x, 'rows', 'first')
hasDuplicates = size(u,1) < size(x,1)
ixDupRows = setdiff(1:size(x,1), I)
dupRowValues = x(ixDupRows,:)

s_features.ins_joints.timestamps[:2].duplicated()
```

Appendix A

Appendix

Include the src files?