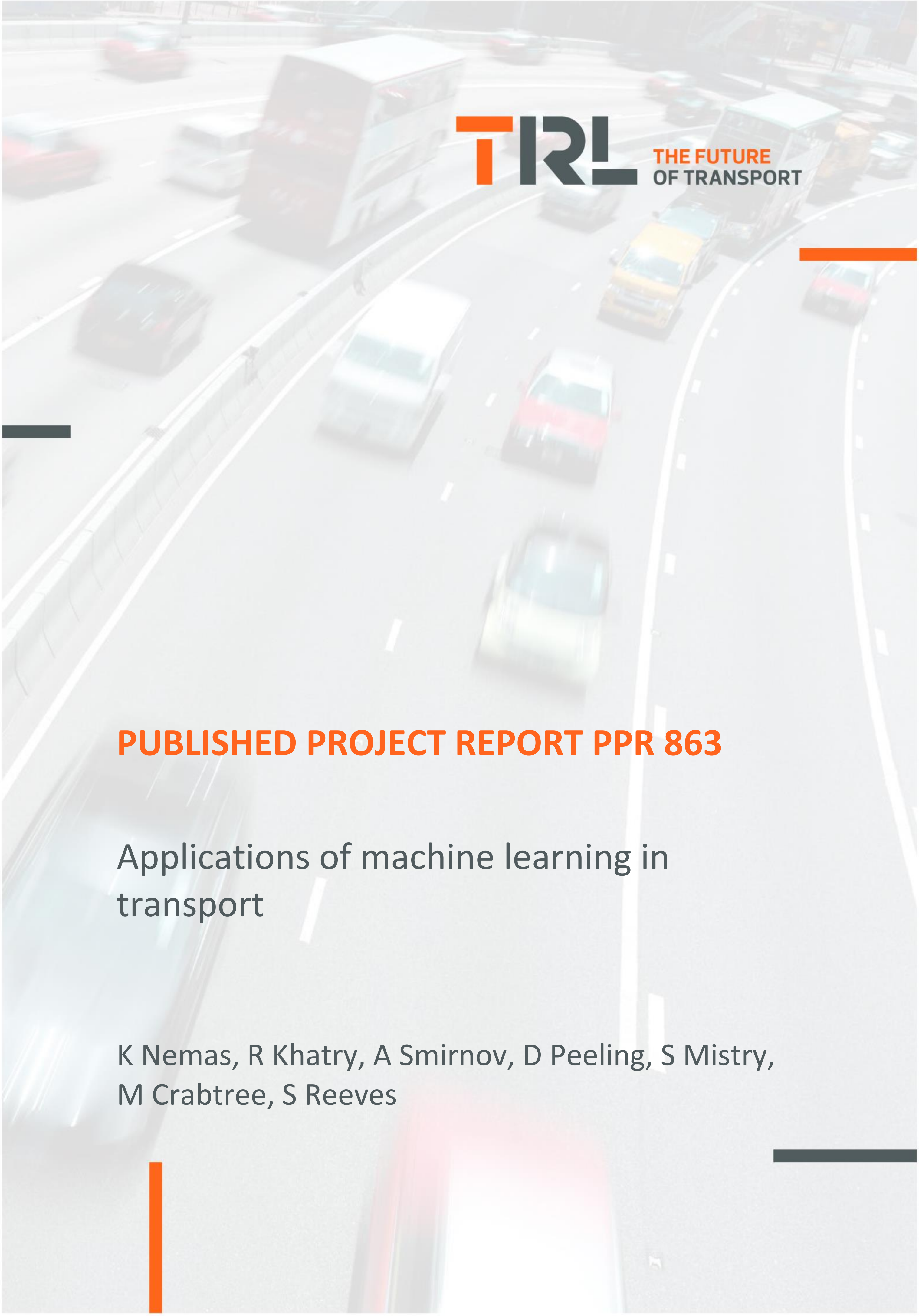


THE FUTURE OF TRANSPORT

PUBLISHED PROJECT REPORT PPR 863

Applications of machine learning in transport

K Nemas, R Khatry, A Smirnov, D Peeling, S Mistry, M Crabtree, S Reeves

Report details

Report prepared for:	TRL Academy		
Project/customer reference:			
Copyright:	© TRL Limited		
Report date:	30 June 2018		
Report status/version:	1.0		
Quality approval:			
Sarah Reeves (Project Manager)		Mark Crabtree (Technical Reviewer)	

Disclaimer

This report has been produced by TRL Limited (TRL) under a contract with TRL Academy. Any views expressed in this report are not necessarily those of TRL Academy.

The information contained herein is the property of TRL Limited and does not necessarily reflect the views or policies of the customer for whom this report was prepared. Whilst every effort has been made to ensure that the matter presented in this report is relevant, accurate and up-to-date, TRL Limited cannot accept any liability for any error or omission, or reliance on part or all of the content in another context.

When purchased in hard copy, this publication is printed on paper that is FSC (Forest Stewardship Council) and TCF (Totally Chlorine Free) registered.

Contents amendment record

This report has been amended and issued as follows:

Version	Date	Description	Editor	Technical Reviewer
0.1	04/05/18	Report structure and outline of contents	Sarah Reeves	Mark Crabtree

Document last saved on:	11/03/2019 14:43
Document last saved by:	Sarah Reeves

Table of Contents

Glossary

Term	Definition
Clustering	Finding important groups in data that have similar characteristics
Decision tree	A tree like structure where different branches mean different outcomes based on input values
Machine learning	Technique for learning patterns from data
Model training	The process of learning model parameters given a data set call training data set
Gradient Boosted	A technique in decision tree which uses multiple models and improves accuracy using learning gradient; where gradient is the difference in accuracy between estimated models
Convolution layer	The layers that are used in a neural network to learn features from the input data.
Max pooling	It is inserted between convolution layers, and is used to selectively pay attention to different sections in an image based on their score.
CNN architecture	The arrangement of layers in a Convolutional Neural Network.
Overfitting	When the learned function is too closely approximating the training data including the unimportant trends and hence becomes less useful for prediction.

Abstract

As the management and analysis of large and complex data sets is becoming an increasingly important part of TRL's work across business areas, the TRL Academy is funding a programme of research that aims to develop and enhance machine learning. Machine learning can process large amounts of data more quickly and efficiently than manual analysis, and it may also reveal relationships between data sets that have not been considered. This report summarises the research carried out under the TRL Limited reinvestment scheme. The project considered three different types of potential applications of machine learning: Analysis of train driver behaviour (clustering), which successfully used clustering to analyse a small data set and still provide some useful conclusions; Condition forecasting of road pavements which showed that the resources of existing data sets is significant and though the final results were inconclusive the act of researching this data has built a stable framework on which to progress; and the crack detection study showed that some of the more mundane and labour intensive processes can be automated and useful results obtained.

Executive summary

Introduction

As the management and analysis of large and complex data sets is becoming an increasingly important part of TRL's work across business areas, the TRL Academy is funding a programme of research that aims to develop and enhance machine learning. Machine learning can process large amounts of data more quickly and efficiently than manual analysis, and it may also reveal relationships between data sets that have not been considered. This report summaries the research carried out under the 2017/18 reinvestment project '*Developing and enhancing Machine Learning skills within TRL'*.

The objectives of the project were to:

- To explore different applications of machine learning across different areas of transport research;
- To develop TRL's capability in machine learning in order to better support clients in its research and consultancy work; and
- To develop a cross-organisational machine learning community in order to exchange ideas and lessons learnt in order to develop the company's skills.

The project considered three different types of potential applications of machine learning in order to fulfil the projects' objectives. The three case studies covered different types of data and analysis techniques but the team met regularly to discuss the work and exchange ideas.

What is machine learning

Machine learning is a mathematical discipline that helps to optimise a performance criterion using example data or past experience. It is becoming more widely used primarily because of the availability of large data bases and greater computing power. The most common applications for machine learning are categorical analysis and image analysis, but it can potentially be used on any kind of dataset, whether the data is numeric, labelled, or mixed.

Types of techniques

Due to the recent popularity of machine learning a variety of tools exist. The related algorithms and programs can be allocated to three areas:

- *Classification* – categorisation of input data into a number of pre-defined groups. Often classification problems are binary, e.g. between two options.
- *Regression* – fitting the data to represent an established type of mathematical function. Generally both linear and non-linear regression is possible.
- *Clustering* – separation of data into a set number of groups (clusters) based on an appropriate measure of distance between data points.

Vast streams of data are generated in the transport industry of varying kinds. Machine learning could be used to process the data into something meaningful and potentially useful. In the context of this project:

- Telemetry data from trains has been used to characterise driver behaviour,
- surveying results have been used to estimate the condition of the road surface, and
- images produced by pavement scanning have been used to the to identify cracks in the road.

Case study 1: Analysis of train driver behaviour (clustering)

Case study 1 considers data from sensors on a train during 20 journeys and 9 drivers between Dublin and Maynooth in Ireland. Examples of the data collected included: speed; brake demand; aspects of CAWS (Continuous Automatic Warning System); aspects of door operation; and power control.

For various reasons it was necessary to make the data more uniform before use. Discrete Time Warping was used to map elements of one time series onto another. The amount of data available was not large, but was sufficient to consider some machine learning principles.

A number of hypotheses were tested and two were found to give coherent results. These were engine power and braking profile. An addition was to see if a particular driver X was significantly different from the general picture. Attempts were made to cluster journeys based on selected data. Some clustering was observed indicating had there been more data available this approach may have given some meaningful results. The study also highlighted how important the data structure is in machine learning in order to produce worthwhile results.

Case study 2: Condition forecasting of road pavements

Forecasting pavement condition is necessary to allow road management authorities to better estimate when a road will require maintenance. Existing algorithms are known to have their limitations and are not trusted by the industry.

Now, the enormous amounts of condition data available can be used to better predict how the pavement will deteriorate. This case study applied a gradient boosted decision tree machine learning algorithm to a set of road surface data with the aim of establishing correlations between survey results and the specific road properties. The decision tree working principle is splitting the data based on the entropy gain or on a similar criterion.

After attempting other methodologies, it was decided to train a model tree for each section, and then design a suitable novel method to have each tree “vote” on the input data. The vote weights were distributed based on the measure of accuracy of that tree.

Thus far the results have not been very useful with error metrics changed wildly over the data range. The analysis was also time consuming: analysing all 26645 sections, including both training and testing, took the order of two working days. One way to save time in future tests would be to ensure that very precise research questions are asked and to conduct more trials in different combinations.

Further work could include linking the pavement condition dataset with other datasets, such as weather and traffic. Doing so could allow predict the deterioration of pavement in more robust and accurate ways.

Case study 3: The use of image processing for detecting cracks and other features in road pavements (classification)

This case study used images from downward and forward facing images of the road pavement. The overall outline of the case study was to use machine learning algorithms for the image processing task. Much of the data had already been processed manually and therefore was one potential source for training and evaluating the success of the algorithms.

The input data was prepared as a Convolutional Neural Network (CNN). Then the effect of varying the different parameters in the CNN, like number of layers, size of filters, learning rate and data batch size was considered as was the effect of different architecture on training and prediction.

The parametric investigation showed that there is still much to explore in terms of developing the technique to achieve the required results. However, even though the methodology required a lot of tweaking to find the right parameter values, the work showed that labour intensive processes can be automated and produce useful results.

Conclusion

These case studies have highlighted that machine learning can be used across the whole transport industry even if the results vary vastly depending on the problem being tackled.

The train driver behaviour study successfully used clustering to analyse a small data set and still provide some useful conclusions.

The condition forecasting of road pavements study shows that the resources of existing data sets is significant. The final results were inconclusive but the act of researching this data has now built a stable framework on which to progress future work.

The crack detection study showed that some of the more mundane and labour intensive processes can be automated and useful results obtained.

Overall, the studies show that machine learning can be applied to numerous areas of the transport industry to help analyse known problems and build frameworks for future research. No one method fits all problem types so careful initial analysis of individual problems is needed to identify the most suitable approach. The transport industry already has a vast array of data sets waiting to be explored and exploited, some of them could even be linked together to enrich the information contained within. Even in those areas where data is scarce it is still possible to identify trends and recommend how to proceed further.

1 Introduction

As the management and analysis of large and complex data sets is becoming an increasingly important part of TRL’s work across business areas, the TRL Academy is funding a programme of research that aims to develop and enhance machine learning skills within TRL. Machine learning is a useful data analysis tool as it can be used to process large amounts of data more quickly and efficiently than manual analysis, and it may also reveal relationships between data sets that have not been considered by researchers. This report summaries the research carried out under the 2017/18 reinvestment project ‘*Developing and enhancing Machine Learning skills within TRL’*.

1.1 Project objectives

The objectives of the project were to:

- To explore different applications of machine learning across different areas of transport research;
- To develop TRL's capability in machine learning in order to support its research and consultancy work; and
- To develop a cross-organisational machine learning community where those with an interest in machine learning can exchange ideas and lessons learnt in order to develop their skills.

The project builds on the successful machine learning project carried out in 2016/17 developing machine learning algorithms to improve TRL’s traffic management software and previous TRL Academy projects related to big data and data analytics.

1.2 Project scope

The project selected three different types of potential applications of machine learning in order to explore the potential benefits of machine learning and grow the team’s capability in different types of machine learning. These three case study examples cover different types of data and analysis techniques. Different members of the team worked on different cases, but all the team met regularly to discuss the case studies and exchange ideas.

1.3 Report layout

The report is divided into the following Sections:

- Section 1 (this section) introduces the project and its objectives
- Section 2 provides a short description of what machine learning is and the various types of methods that can be employed.
- Sections 3, 4 and 5 describe the three case studies including the research objectives, method employed, results and conclusions.
- Section 6 discusses what has been learnt from the three case studies and potential further work.

- Section 7 summaries the project findings and conclusions.

2 Background

2.1 What is machine learning

Machine learning is a mathematical discipline that helps to optimise a performance criterion using example data or past experience. A good machine learning algorithm organises the available data in the best way possible. The “best” here means that the interaction of the real life objects that produced that data in the first place can be replicated, and the new incoming data can be allocated in a similar way. Of course, this means that the outcome of machine learning mostly depends on how representative the input data is of a typical situation. Usually, that is ensured by collecting a sufficiently large amount of data, which makes machine learning impractical to do manually and warrants utilising large amounts of computing power.

The most common applications for machine learning are categorical analysis and image analysis, but it can potentially be used on any kind of dataset, whether the data is numeric, labelled, or mixed. The biggest challenge lies in collecting enough data points to adequately train a model. In this context, a “model” is an assumed relationship between the inputs and the outputs, while “training” is changing that model until it is true for the largest proportion of the dataset possible.

2.2 Types of techniques

Due to the recent popularity of machine learning in almost every industry that handles large volumes of data, a variety of tools exist for both the layman and professional. Roughly, the related algorithms and programs can be allocated to three areas:

- *Classification* – categorisation of input data into a number of pre-defined groups. Often classification problems are binary, e.g. between two options, and this class of problems is the most well-researched.
- *Regression* – fitting the data to represent an established type of mathematical function. There are several approaches, but generally both linear and non-linear regression is possible.
- *Clustering* – separation of data into a set number of groups (clusters) based on an appropriate measure of distance between data points.

In the above paragraph, classification and regression are termed “supervised” methods, while clustering is “unsupervised”. “Supervised” machine learning can be defined as any approach that involves somebody outside of the algorithm defining beforehand what data is considered an input or an output. “Unsupervised” machine learning, by contrast, assumes nothing about how the data points relate to each other prior to the machine learning process. Generally, supervised methods are easier and faster to implement and interpret, while the unsupervised methods require less pre-processing.

2.3 The use of machine learning in transport

All modern means of transportation implement telemetry and feedback devices in some capacity, since any driver needs to be aware of the situation around the vehicle. These devices generate vast streams of data, and most of it is discarded after its acquisition and immediate use. Machine Learning could be used to process the data and the amounts of data available would be expected to guarantee that the algorithms would be able to sort and organise the available information into something meaningful and potentially useful. This means the way to handle the situation better could be easily identified. In the context of this project:

- Telemetry data from trains has been used to characterise driver behaviour,
- surveying results have been used to estimate the condition of the road surface, and
- images produced by pavement scanning have been used to the to identify cracks in the road.

3 Case study 1: Analysis of train driver behaviour (clustering)

This case study is based on the results of analysing a set of 40 tables containing readings from various sensors on a train during 20 journeys between stations Connolly to Maynooth and Maynooth to Pearse in Ireland. Nine drivers were registered on these journeys, with driving experience ranging from 2 to 14 years. The number of journeys each driver completed varied as well – drivers 3, 5 and 6 having the most journeys related to them. The forward and return journeys have been analysed separately.

3.1 Objectives

The aim of the case study was to investigate whether algorithmic methods can produce results on par with the estimation abilities of an experienced consultant. Further, in this section elaboration of the data structure was sought, expectations and results of both methods was explained, and conclusions drawn on how the two compare.

3.2 Methodology

3.2.1 Description of data

Each journey is associated with a continuous stream of data recorded in spreadsheet, with each column corresponding to an individual input variable. All available variables were recorded for each instance, some of them being essentially binary flags. Among the recorded variables were:

- Record ID – Unique Identifier.
- Date – Recording date.
- Time – Timestamp.
- Distance – distance counter from the moment of installation until present.
- System speed – roughly accurate speed of travel at the time of recording.
- Brake demand – a value between 0 and 7 associated with the position of the brake control.
- CAWS – aspects of CAWS (Continuous Automatic Warning System).
- Forward – is the train moving forward (1 if true).
- Reverse – Is the train reversing (1 if true).
- Horn switch – is the driver sounding the horn (1 if true).
- Door controls – various aspects of door operation
- Latitude – GPS coordinate.
- Power control - a value between 0 and 5 describing the position of the power control lever.

The data was presented in series to mirror the way it was collected throughout the journey. However, the number of data points varies within each individual file. This was because although the recording triggered on each sensor change, some sensor inputs have been omitted from the available information due to irrelevancy. As a result, it was necessary to make the data more uniform before any calculations could be made. The solution was to use discrete time warping (DTW), an algorithm that permits mapping elements of one time series onto another. To present the data in a standard format, and to avoid interpolating extra data into the system, all files have been warped with respect to the one with the least number of records.

3.3 Analytic estimates

The following hypotheses have been investigated:

1. Drivers can be categorised by using high braking levels
2. Drivers can be categorised by time elapsed between CAWS change and response
3. Journeys can be categorised by time spent at red signals
4. Journeys can be categorised on usage the emergency brake on approach to a red signal
5. Journeys can be categorised on putting the train in neutral when stopped at stations
6. Drivers can be categorised on how they use the horn
7. Drivers can be categorised on time elapsed between doors closing and power application, and also duration of the door opening
8. Drivers can be classified on power application profiles

Previous internal TRL research has shown:

- Drivers can be classified by power reading.
- Drivers can be distinctly classified on braking profiles.
- The time spent between stopping at a station and the doors opening has regularly exceeded the standard recommended time of 30 seconds.
- Door-closing-to-power-applied-time is on average 8 seconds.
- Red signal approach was not a significant factor.
- CAWS reaction was not found to be significant.
- Emergency brake usage is rare but it is reasonable to expect that using an emergency brake before a red light indicates a Signal Passed At Danger (SPAD).
- Initiating and changing power has a positive correlation with heart rate.
- Initiating and changing braking has a negative correlation with heart rate.
- Hypothesis 1 deemed plausible.
- Hypothesis 2 somewhat holds up but is more relevant to journeys as opposed to drivers.
- Hypothesis 4 is worth considering.
- Hypothesis 5 Is false.
- Hypothesis 6 relies on inaccurately recorded data and it is not useful.
- Hypothesis 7 is probably plausible.
- Hypothesis 8 is correct.
- Forward/Reverse settings didn’t have any trends associated with them.

The feasibility study was carried out as preliminary clustering to see if sensible results could be obtained. The hypotheses tested were 1,2,4,7 and 8. The preliminaries indicated that only 1 and 8 were producing coherent results.

The research problem has been formulated to confirm whether the following variables would be the most significant in characterising drivers:

- Engine Power
- Braking profile

An addition was to see if driver X (driver 4 in graphs in Figures 1 to 4) was significantly different from the general picture.

3.4 Results

After pre-processing the files, attempts were made to cluster journeys based on selected columns in the data, treating the whole column as a data vector. Clustering with K-means based on Euclidian distance revealed groupings similar in some aspects to analytical findings. The K = 9 configuration was chosen, because the expected result would be 9 distinct clusters equal to the number of drivers, as a reference to show the quality of the metric and its reliability.

Connolly – Maynooth route (note – each column is a cluster, the red numbers mark the driver who did the journey:

- Engine power – Almost all journeys done by driver 3 are together in the same cluster, except for one. Nothing conclusive can be said about other drivers.

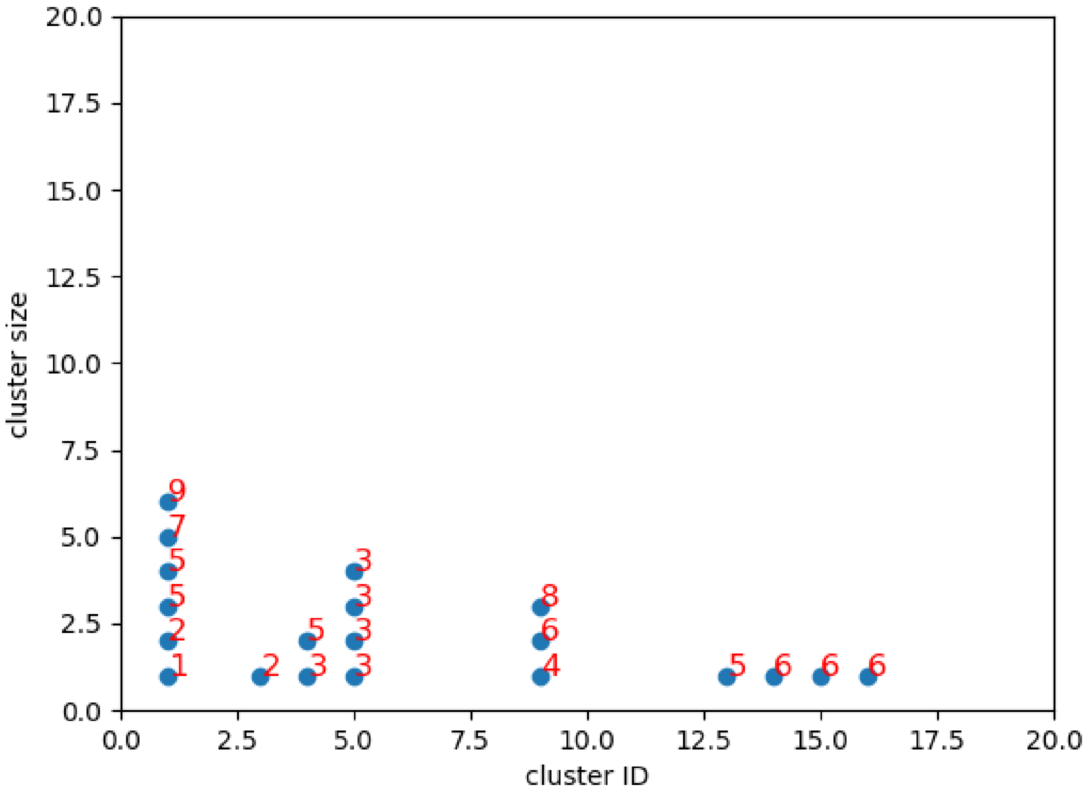


Figure 1.Clustering of the Connolly – Maynooth journeys based on application of the engine power by the driver during the journey.

- Braking power – perfectly clustered driver 3, drivers 5 and 6 now have a cluster each consisting exclusively of that driver’s journeys, which is an improvement compared to clustering based on the engine power.

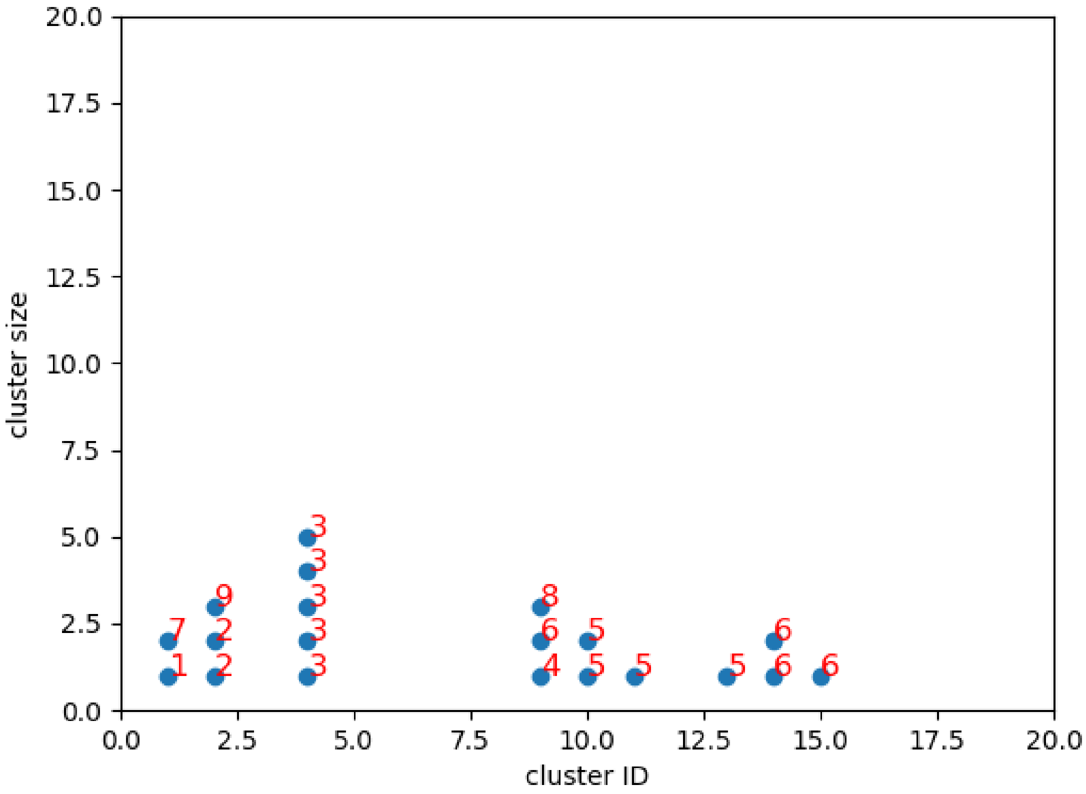


Figure 2. Clustering of the Connolly – Maynooth journeys based on application of the brake power by the driver during the journey

Maynooth to Pearse (return journey, Pearse is a station neighbouring Connolly):

- Engine power – Good results on singling out driver 3, isolated single-journeyed drivers 7 and 8, barely passable results on drivers 5 and 6 because some of their journeys are grouped together. …

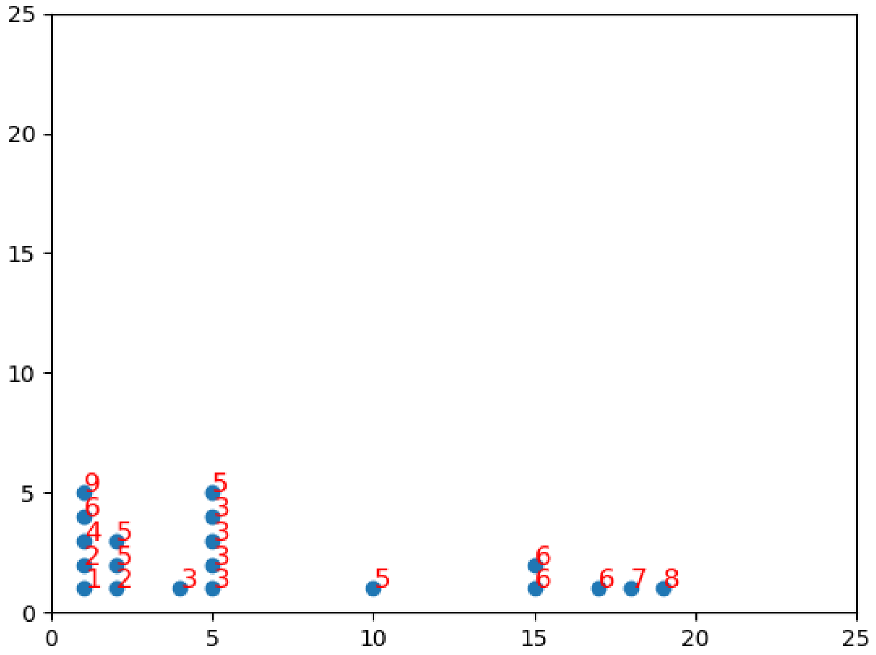


Figure 3. Clustering of the Maynooth - Pearse journeys based on application of the engine power by the driver during the journey.

- Brake power – still separates drivers 7 and 8, but drivers 3 and 6 are on par now in terms of clustering quality.

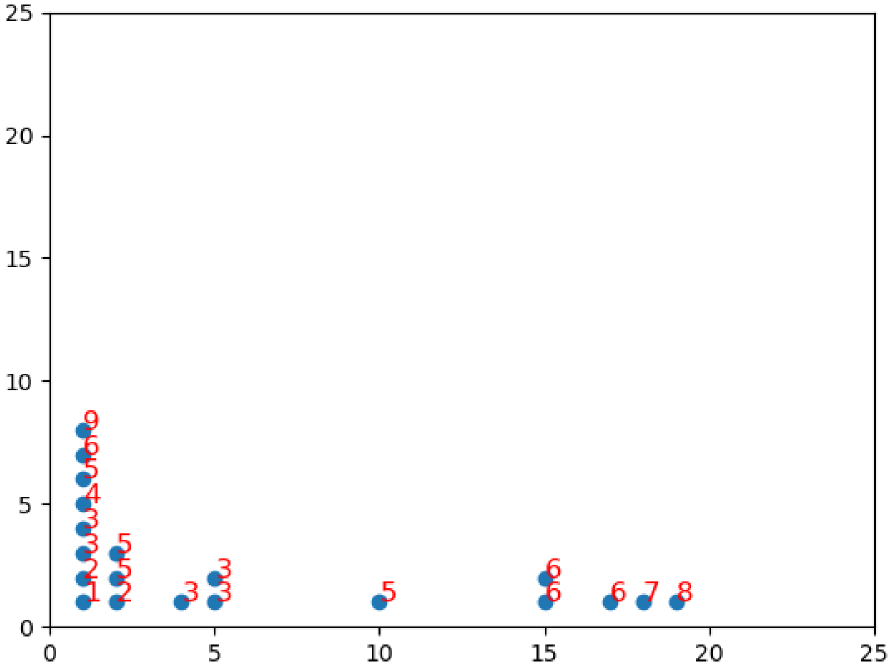


Figure 4. Clustering of the Maynooth - Pearse journeys based on application of the brake power by the driver during the journey

3.5 Conclusions and recommendations

Clustering on some variables was in line with the analytical expectations, but due to the varying number of journeys associated with each driver, and the small amount of data points, it is difficult to conclude unambiguously. In the end, it was possible to conclude that a machine learning approach would be able to hint at correlations similar to those recognised by a human, but making it approach manual observation in quality was not possible in this particular case study due to the limited amount of data. In addition, the fact that there was an uneven number of journeys per driver seriously affected the accuracy of the clustering algorithm. Quite a few drivers have had only one journey associated with them and this further hindered useful observations. This case study highlights how important the data structure is in machine learning in order to produce worthwhile results.

Even with the very limited amount of data it doesn't take too much imagination to appreciate that much could be achieved with much larger amounts of data – amounts though would normally be very easy to collect under similar circumstance.

4 Case study 2: Condition forecasting of road pavements

Forecasting pavement condition is necessary because it can enable road management authorities to better estimate when a road will require maintenance. Being able to do this accurately has many economic and societal benefits. Maintaining the road at the right time can prevent unnecessary interventions that cost more money, cause more roadworks and therefore delays to the public. Existing algorithms used to forecast road pavement condition are known to have their limitations: those currently used by Highways England, for example, are over 15 years old and are not trusted by the industry.

Advances in big data and machine learning techniques make it possible to improve upon existing algorithms. The advantage of using this type of methodology rather than existing methods is that there are enormous amounts of condition data currently available that can be used to better predict how the pavement will deteriorate. It is also possible to link the data to other datasets that will impact upon the condition of the pavement e.g. Traffic.

This case study applied a gradient boosted decision tree machine learning algorithm to a set of road surface data with the aim of establishing correlations between survey results and the specific road properties. The methods used and the steps taken to create a suitable processing environment for that data are described below.

4.1 Objectives

The case study objectives were to:

- Download data from Highways England Pavement Management System (HAPMS) for use in the analysis including network, construction and condition data
- Convert the downloaded data into a PostgreSQL format to enable easier access and processing, while simultaneously filtering out redundant columns.
- Link the Postgres tables together so that the queries would be more straightforward and use numerical identifiers only.
- Design, build and test the Python script which would store the results of training in testing in an accessible way.
- Explore the available data in order to train a predictive model based on correlating variables and later to create a model that would estimate deterioration rates or future survey results.

4.2 Methodology

4.2.1 Approach

Decision trees are the most versatile machine learning tool currently available. Their working principle is splitting the data based on the entropy gain or on a similar criterion. Decision trees deduce conditional relationships, rather than linear dependencies, making them, in principle, easier to understand.

The algorithm used in this instance has been a powerful C/C++ based open-source package XGBoost, a popular tool for most machine learning problems due to its ability to compute both regression and classification models, work with missing values and flexible settings.

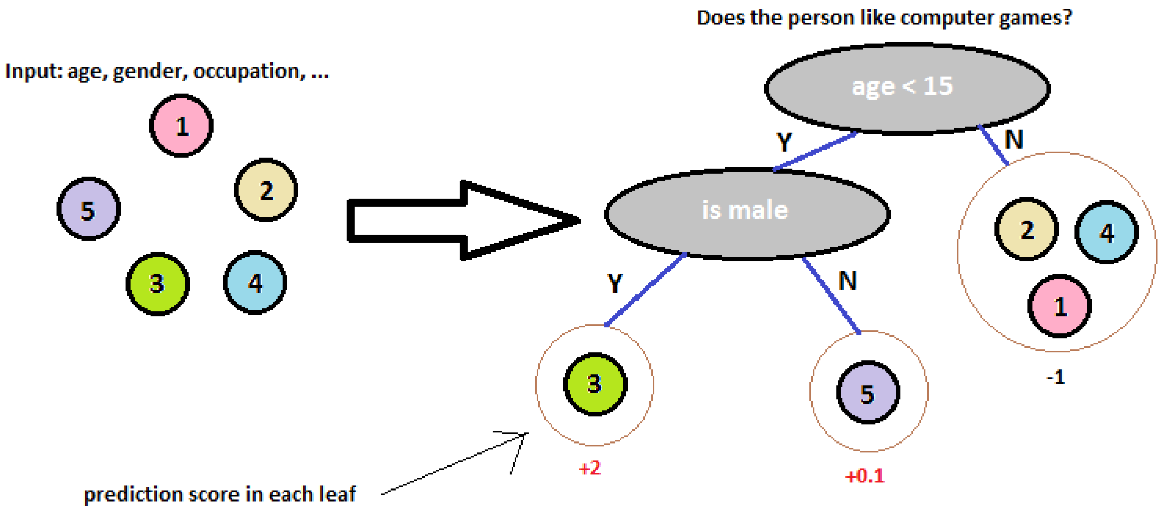

Does the person like computer games?
Input: age, gender, occupation, ...
1
2
5
3
4
age < 15
Y
N
is male
Y
N
2
4
1
-1
3
5
+2
+0.1
prediction score in each leaf

Figure 5 Example of what a decision tree is like

In a Classification And Regression Tree (CART) each leaf on the tree has a score associated with it, which is later used as a weight on a result.

For large and convoluted sets of data, one tree might not be enough. In such cases XGBoost uses an ensemble of CARTs that are split in a multitude of ways initially and thus end up writing complementary scores for the data. These scores can be combined and compared across different trees for each item, which means every individual result is an aggregate.

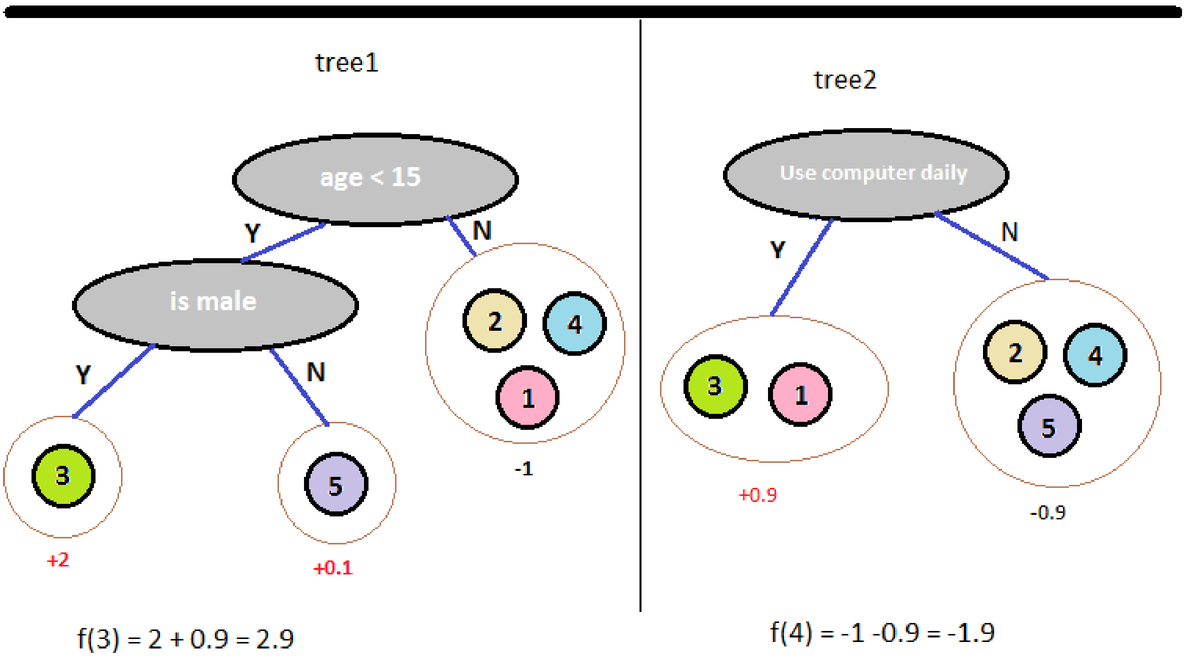


Figure 6 “Decision forest”.

XGBoost trees are called “boosted”, because each learning iteration (except for the first one) is based on the previous objective function. This “nudges” the next model in the right direction.

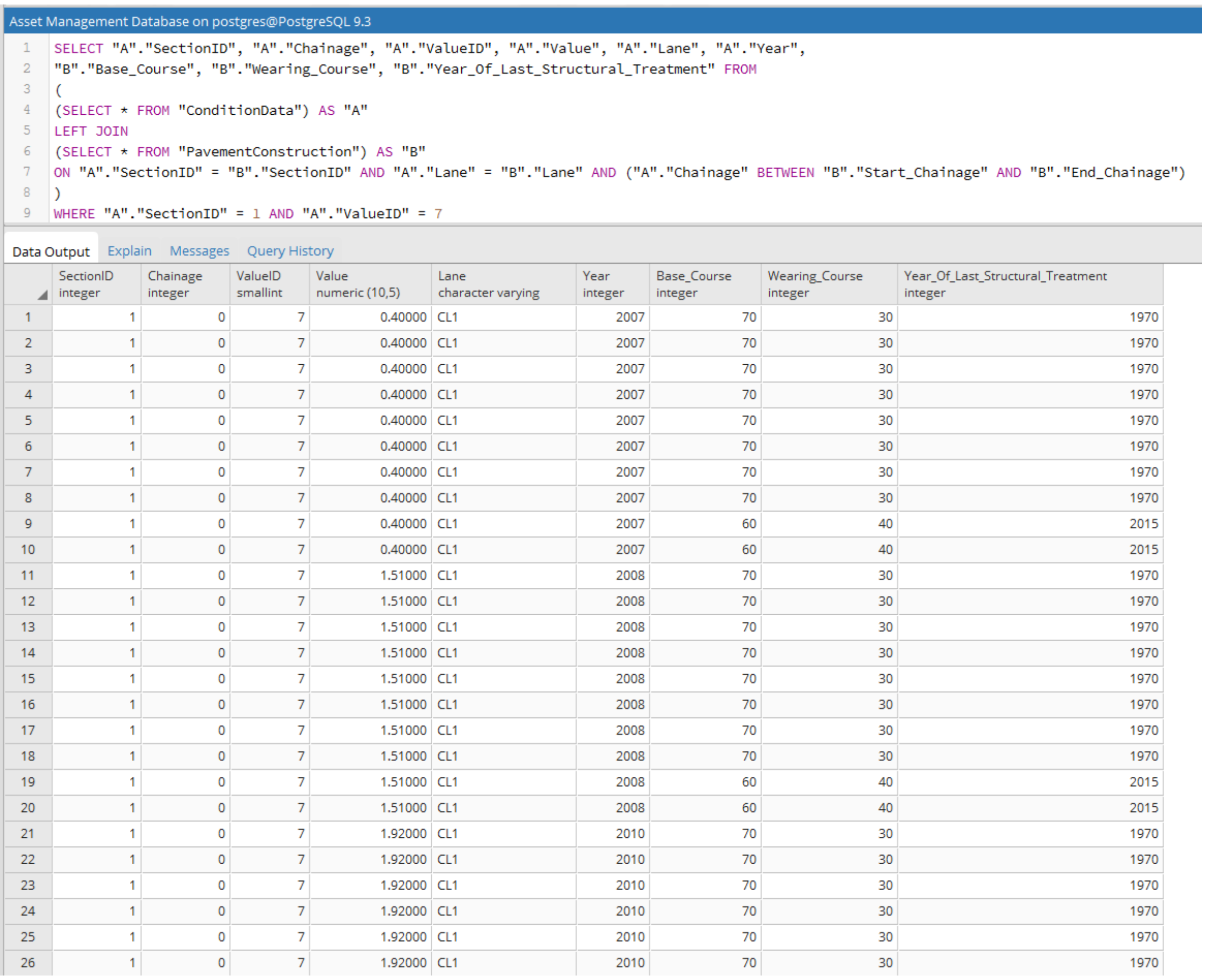
4.2.2 Data set

The following datasets were downloaded from HAPMS as of 31st March 2017 and dating back 10 years:

- Network Data
- Pavement Condition Data (TRACS)
- Pavement Construction Data

The pavement condition dataset are results of the TRAffic-speed Condition Surveys (TRACS) that include surface defects such as the transverse and longitudinal profile of the road. The data is referenced by the HAPMS road sections data that also links to the pavement construction data.

Figure 7 shows the typical rows of data generated by a single request in PostgreSQL format.



```sql
SELECT "A"."SectionID", "A"."Chainage", "A"."ValueID", "A"."Value", "A"."Lane", "A"."Year",
"B"."Base_Course", "B"."Wearing_Course", "B"."Year_Of_Last_Structural_Treatment" FROM
(
(SELECT * FROM "ConditionData") AS "A"
LEFT JOIN
(SELECT * FROM "PavementConstruction") AS "B"
ON "A"."SectionID" = "B"."SectionID" AND "A"."Lane" = "B"."Lane" AND ("A"."Chainage" BETWEEN "B"."Start_Chainage" AND "B"."End_Chainage")
)
WHERE "A"."SectionID" = 1 AND "A"."ValueID" = 7
```

	SectionID integer	Chainage integer	ValueID smallint	Value numeric (10,5)	Lane character varying	Year integer	Base_Course integer	Wearing_Course integer	Year_Of_Last_Structural_Treatment integer
1	1	0	7	0.40000	CL1	2007	70	30	1970
2	1	0	7	0.40000	CL1	2007	70	30	1970
3	1	0	7	0.40000	CL1	2007	70	30	1970
4	1	0	7	0.40000	CL1	2007	70	30	1970
5	1	0	7	0.40000	CL1	2007	70	30	1970
6	1	0	7	0.40000	CL1	2007	70	30	1970
7	1	0	7	0.40000	CL1	2007	70	30	1970
8	1	0	7	0.40000	CL1	2007	70	30	1970
9	1	0	7	0.40000	CL1	2007	60	40	2015
10	1	0	7	0.40000	CL1	2007	60	40	2015
11	1	0	7	1.51000	CL1	2008	70	30	1970
12	1	0	7	1.51000	CL1	2008	70	30	1970
13	1	0	7	1.51000	CL1	2008	70	30	1970
14	1	0	7	1.51000	CL1	2008	70	30	1970
15	1	0	7	1.51000	CL1	2008	70	30	1970
16	1	0	7	1.51000	CL1	2008	70	30	1970
17	1	0	7	1.51000	CL1	2008	70	30	1970
18	1	0	7	1.51000	CL1	2008	70	30	1970
19	1	0	7	1.51000	CL1	2008	60	40	2015
20	1	0	7	1.51000	CL1	2008	60	40	2015
21	1	0	7	1.92000	CL1	2010	70	30	1970
22	1	0	7	1.92000	CL1	2010	70	30	1970
23	1	0	7	1.92000	CL1	2010	70	30	1970
24	1	0	7	1.92000	CL1	2010	70	30	1970
25	1	0	7	1.92000	CL1	2010	70	30	1970
26	1	0	7	1.92000	CL1	2010	70	30	1970

Figure 7. Example data set

There are 26645 road sections in total, and the data has been collected over 17 years (2000 - 2017) leading to about 20 GB worth of data. This dramatically increases the time it takes to process the model unless it is possible to discard some of the data. There are 17 variables in total, 10 of them are the survey results and 7 are the pavement and construction data.

4.2.3 Calculations

Due to the size of the dataset and the memory limitations, the data, which was kept on a central database, was pre-processed locally on the machine. For the initial experiments, in order to estimate the future timescales, regression on 16 variables to 1 was attempted, the output variable being “texture” of the road. Initially, the combined model was trained on all sections at once, but this design quickly proved to be very error prone and too slow on the scale of a day. It was decided to train a model tree for each section, and then design a suitable novel method to have each tree “vote” on the input data. The vote weights were distributed based on the measure of accuracy of that tree. The first experiment was merely an attempt to validate the decision tree regressor.

4.3 Results

The results from analysing the “texture” for the first 10,000 sections are shown below. The variable on the first diagram – “explained variance score”, reflects how much the predicted variance was explained by the variance in real data:

$\left(1 - \frac{Var\{y-\hat{y}\}}{Var\{y\}}\right), y\ is\ the\ true\ value, \hat{y}\ is\ the\ prediction.$

The second diagram plots a logarithm of Root Mean Square Error:

$$\log_{10}\sqrt{\frac{1}{n}\sum(y-\hat{y})^2}, n\ is\ the\ number\ of\ samples.$$

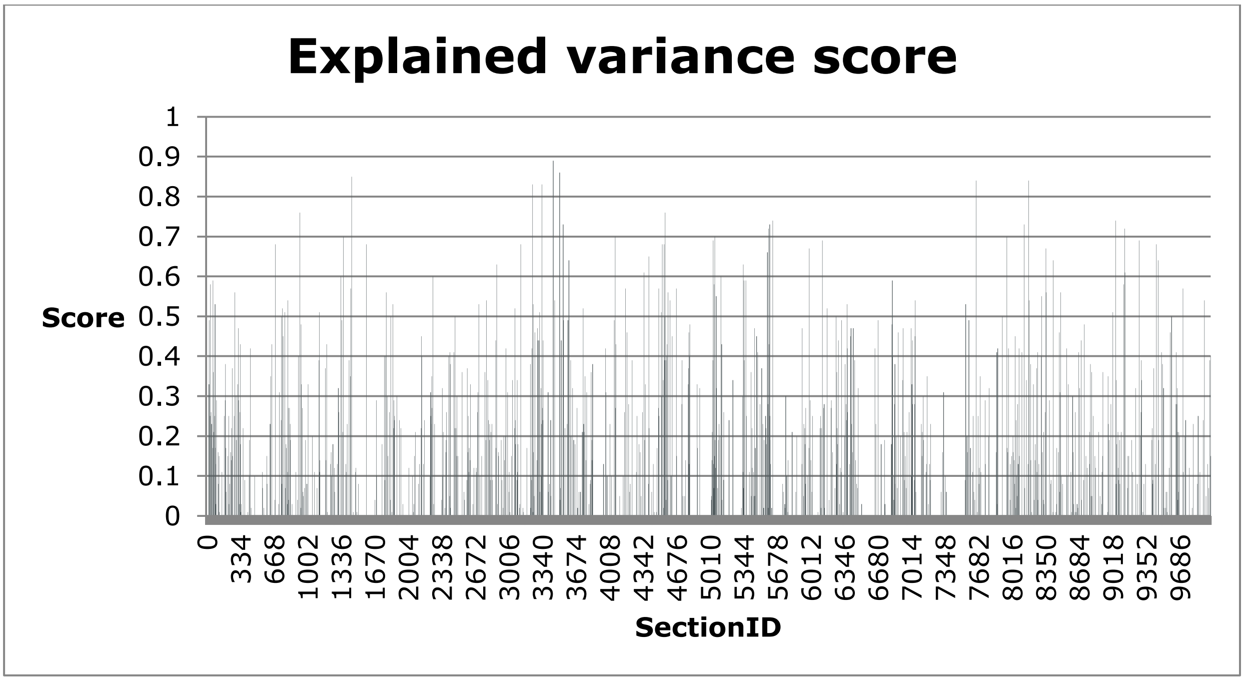


Figure 8. Explained variance calculated for each section of the network based on the assumed relationship between the texture parameter and other 9 parameters of the survey.

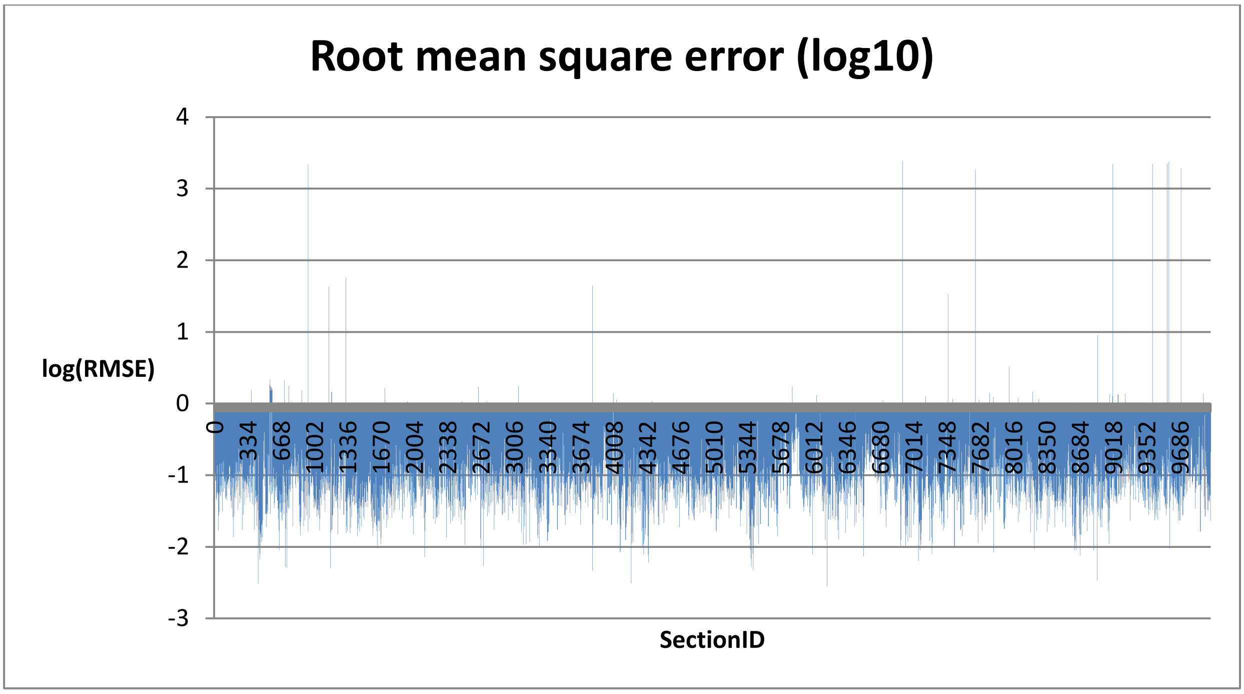


Figure 9. Log of the error in predicting texture of the road for each section based on the assumed model.

Thus far the results have not been very helpful: both error metrics changed wildly over the data range. The analysis was also time consuming: analysing all 26645 sections, including both training and testing, took the order of two working days. One way to save time in future tests would be to ensure that very precise research questions are asked and to conduct more trials in different combinations.

4.4 Conclusions and recommendations

So far the project has resulted in a stable data retrieval system implemented as a python script. This made it possible to quickly query the database to collate any number of inputs and outputs from those available and produce a single global model for the dataset or a multitude of section-specific ones. The script also allowed for the assessment of individual models, and could perform a combined aggregate test on several trees at once. Further work could include designing a procedure that would provide consistently good estimates for at least one of the outputs.

Further work could also include linking the pavement condition dataset with other datasets, such as weather and traffic. These datasets already exist and linking them using krigging and interpolation methods could allow the deterioration of pavement to be predicted in more robust and accurate ways. This work would require some significant processing and time, however, because of the finer temporal granularity of weather and traffic in comparison to pavement condition.

5 Case study 3: The use of image processing for detecting cracks and other features in road pavements (classification)

The work described in this case study focuses on image analysis, complementing the other case studies which focused on the analysis of numerical data. This case study focused on the analysis of images of road pavements collected through Surface Condition Assessment of National Network of Roads (SCANNER) consisting of downward and forward facing images of the road pavement. Much of the data had already been processed manually and therefore was one potential source for training and evaluating the success of the algorithms.

5.1 Objectives

The overall outline of the case study was to use machine learning algorithms for the image processing task. The main objective of the work was to develop TRL's capability in machine learning, and additionally identify an approach for detecting or assessing defects from images, and developing ideas on how the approach could be developed further for analysing other image sets for example forward facing images.

5.2 Methodology

In this section the preparation of the input data to the Convolutional Neural Network (CNN) is discussed, then the effect on the accuracy of varying the different parameters in CNN like number of layers, size of filters, learning rate and data batch size (also called hyper-parameters to distinguish from network parameters like weights and biases that are learned is the training stage) and the effect of different architecture on training and prediction was investigated.

5.2.1 Data preparation

The input data to the CNN network was raw images of the road surface. The raw image was split into sub-images of 200mm × 200mm in dimension, then each sub image labelled "Yes" if it contained a feature for example a crack or an iron work (see Figure 10 and Figure 11), and "No" if not.

The data set was split randomly into two halves, a training set and a validation set. The first set was used to train the machine learning algorithm, and the second set to assess its predictive accuracy.

Supervised training of the machine learning was used, in the sense that the training was performed on a data set for which the type of defect to classify was known.

The data preparation stage included any pre-processing applied to the images. This included aligning the image to the manual grid as depicted in Figure 10 and Figure 11 for the case of cracking and iron work respectively. Other processing included enhancing the image quality to reduce the effect of lighting and increasing the contrast of the features.

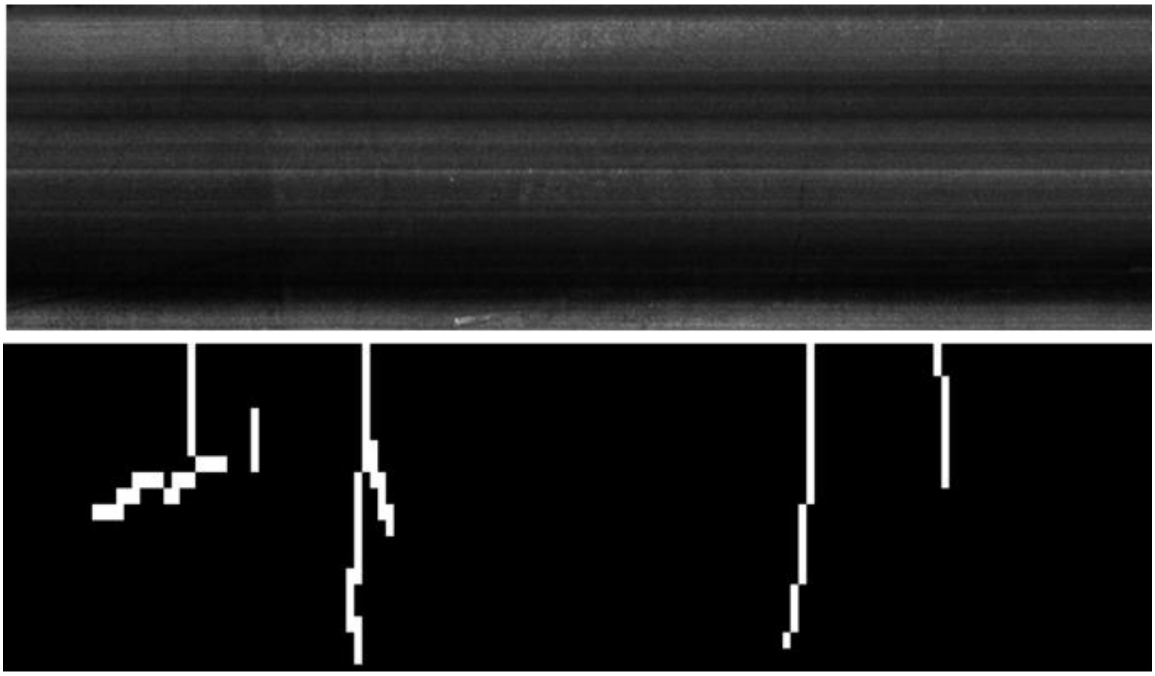


Figure 10: Aligning the road image (Top) and the manual analysis image (Bottom) showing cracking

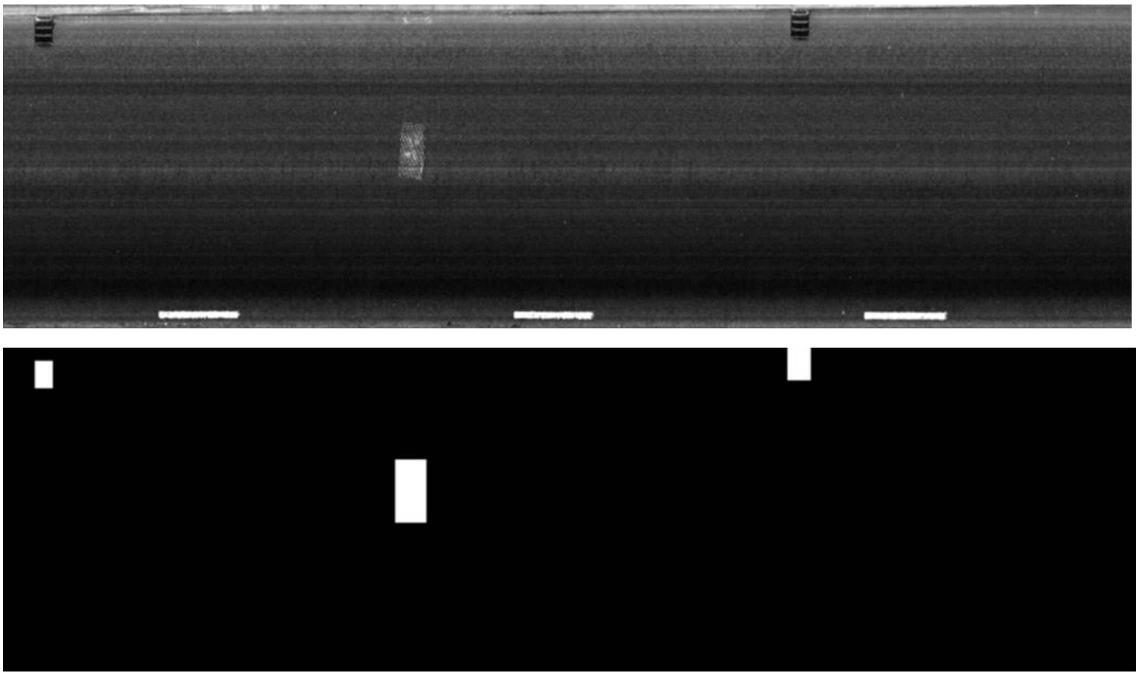


Figure 11: Aligning the road image (Top) and the manual analysis image (Bottom) showing iron work

A data augmentation procedure was performed using transformations depicted in Figure 13. Additional transformations (not shown in the figure) that involved rotation of the image with angles of 30^o, 45^o, and 120^o and expansion were also used. The data augmentation used in the case of cracking, resulted in approximately 14% of the sub-images having a crack in them; and 86% not having a crack.

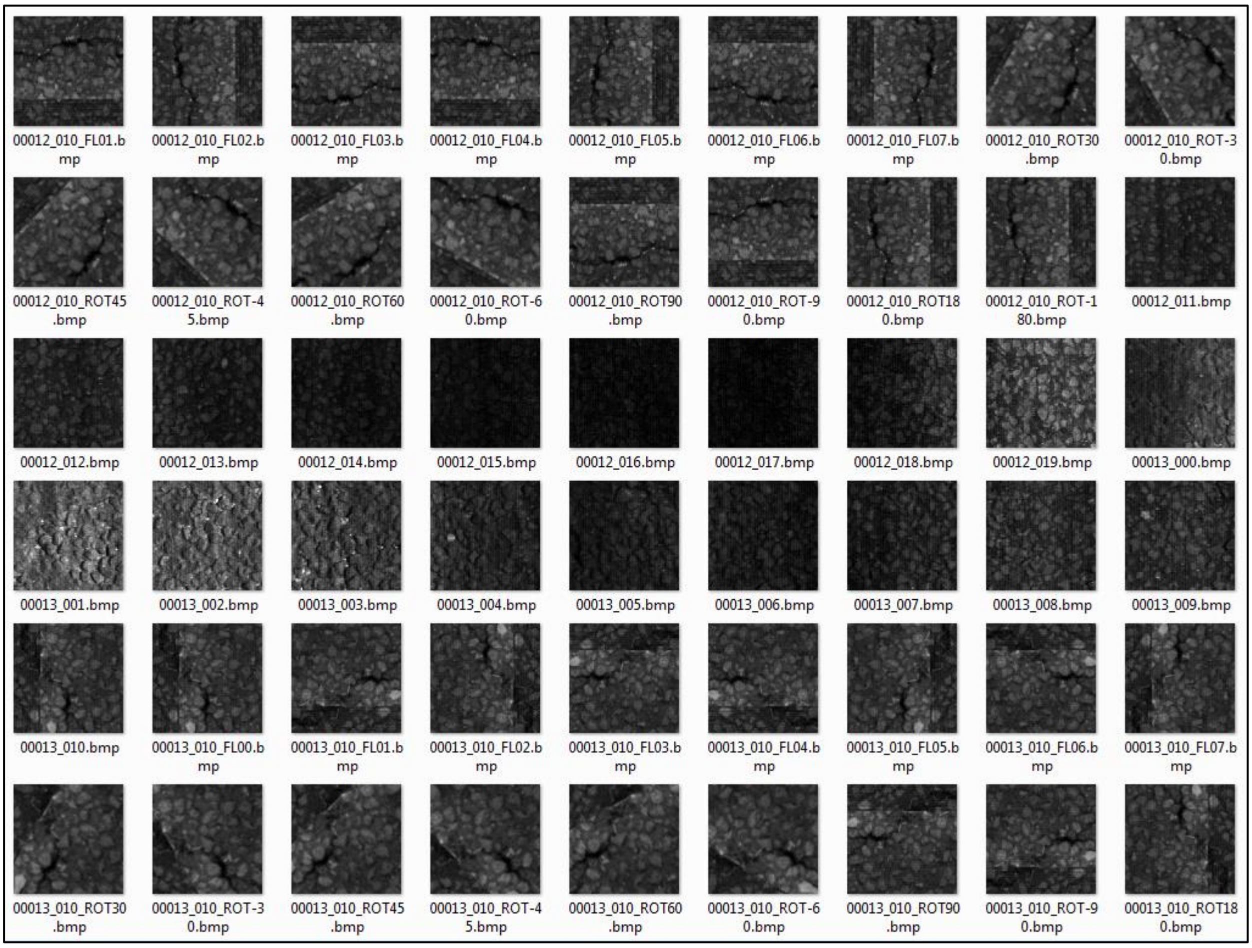
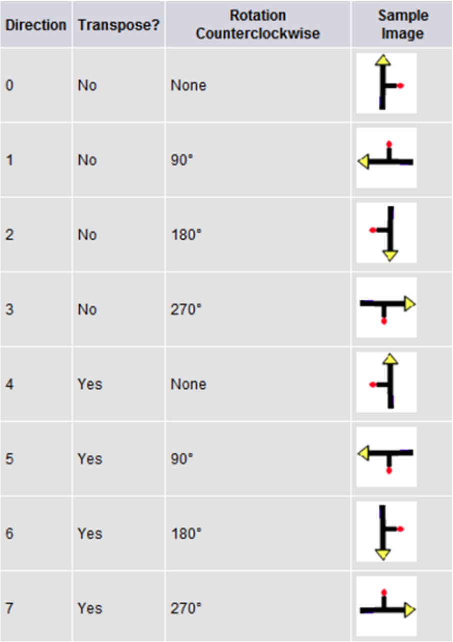


Figure 12: A sample of Sub-images used for training a Convolutional Neural Network, the sub-images that contain a crack were



Direction	Transpose?	Rotation Counterclockwise	Sample Image
0	No	None	
1	No	90°	
2	No	180°	
3	No	270°	
4	Yes	None	
5	Yes	90°	
6	Yes	180°	
7	Yes	270°	

Figure 13: Transformation applied to the sub-images for data augmentation

5.2.2 Model description

The different kinds of components of a convolutional neural network are convolution layers, max-pooling layers, fully connected layers.

The convolutional layer, basically applies filters to the image in order to learn semantic features from it, then uses these learned features to assign scores to different parts of an image. The max pooling layer reduces the size of the images to highlight only the sections associated to the important features and ignore the unimportant regions. The fully connected layer takes the scores and classifies the images into different classes based on the scores gives for different features.

The CNN architecture had six convolution layers with a max pooling layer inserted after each convolution layer. Dropouts were added after each two convolutions and after the fully connected layer, see Figure 14.

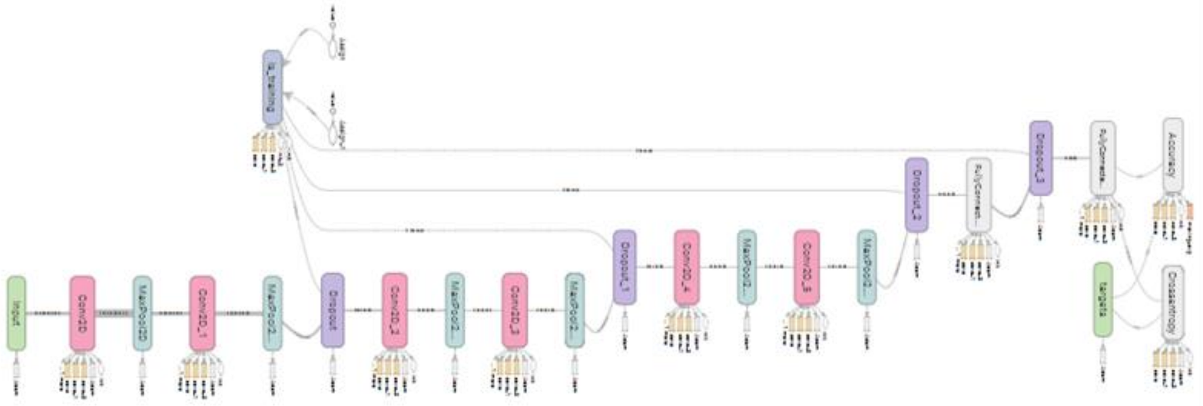


Figure 14: CNN architecture used in this study

5.2.3 Training approach

The training data set was split into a training data set and validation data set. The size of the validation set was 6% of the size of the training set, about 5,000 images, still enough to give an idea about the accuracy of the classifier.

5.3 Results

The assessment of the performance started first by investigating the effect of varying some parameters of the CNN on the accuracy to assess the performance of the classifier

This was followed by calculating the classification errors on the validation data set in section 5.3.2, and the false positives and false negatives arranged in the form of a matrix known as the confusion matrix to show the percentage of genuine defects.

5.3.1 Parametric investigation

5.3.1.1 Effect of batch size

Batch size controls the amount of data accessed in any iteration. Figure 15 depicts the results of accuracy for batch sizes of 32, 64, 128, and 256, and Figure 16 depicts the loss. It is observed that a batch size of 32 yielded the lowest accuracy and the highest loss. The results show that the higher the batch size, the better the accuracy and the lower the loss.

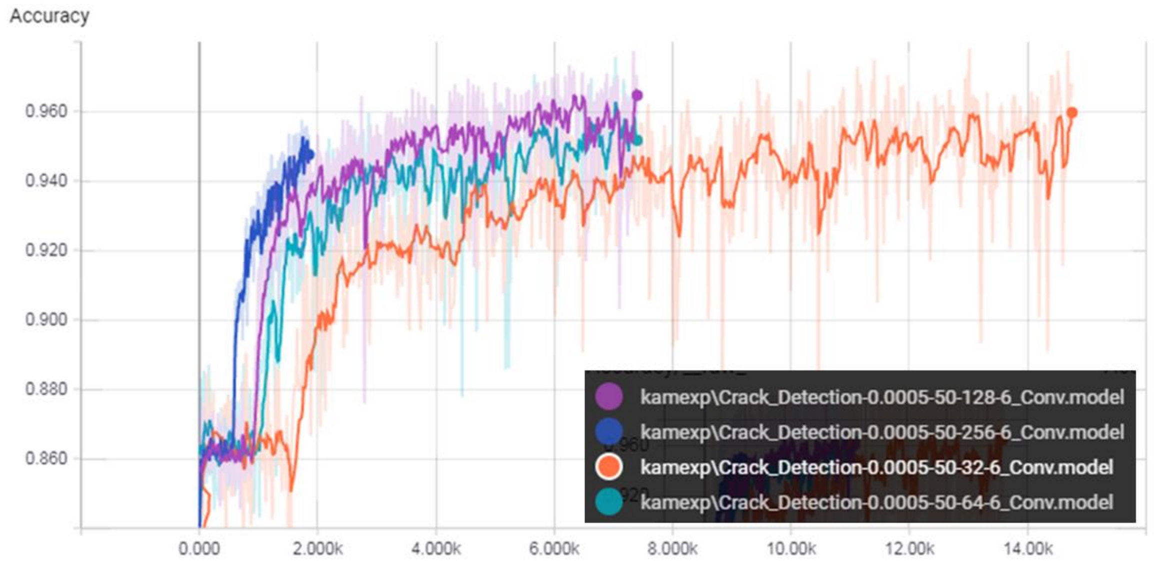


Figure 15: Effect of batch size on the Accuracy curve

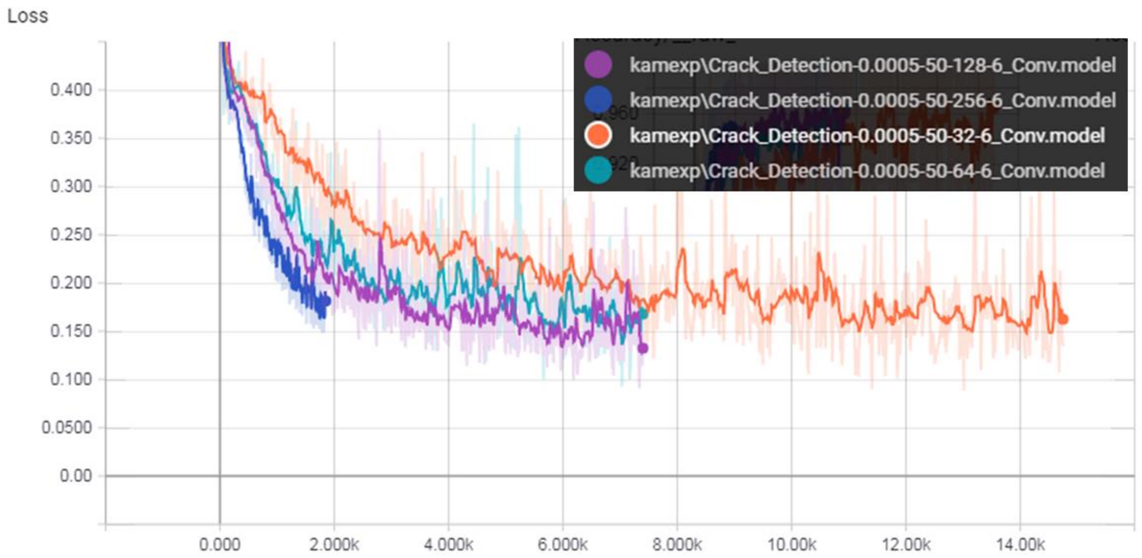


Figure 16: Effect of batch size on the loss curve

5.3.1.2 Effect of learning rate

The learning rate was varied as 0.0005, 0.001 and 0.005. It was observed that the learning rate affected the accuracy and the loss. For instance increasing the learning rate from 0.001 to 0.005 resulted in a decrease of the accuracy of classification of images by approximately 2.4%. The loss increased by approximately 25%.

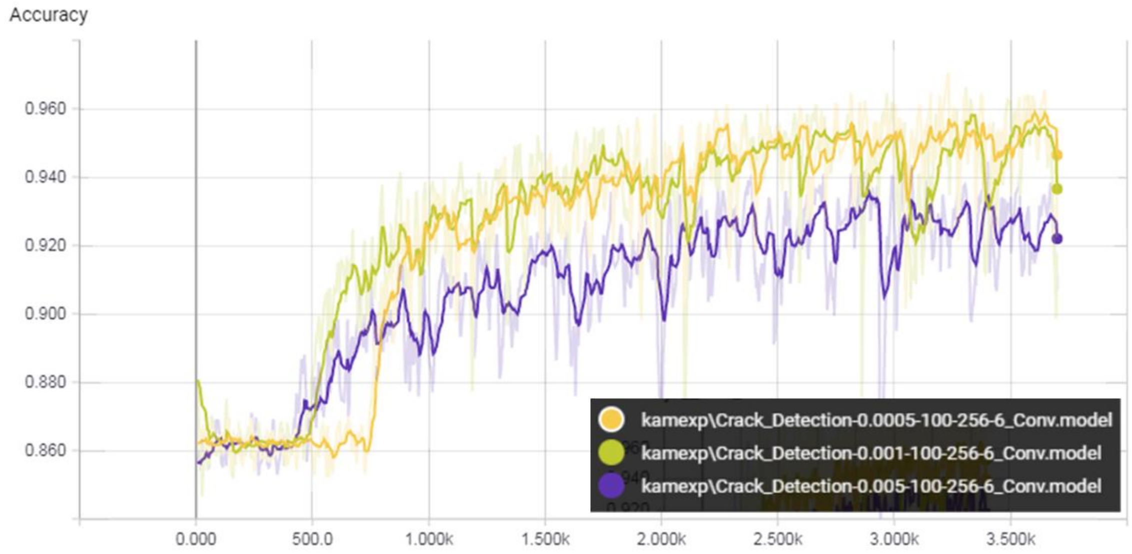


Figure 17: Effect of learning rate on the accuracy

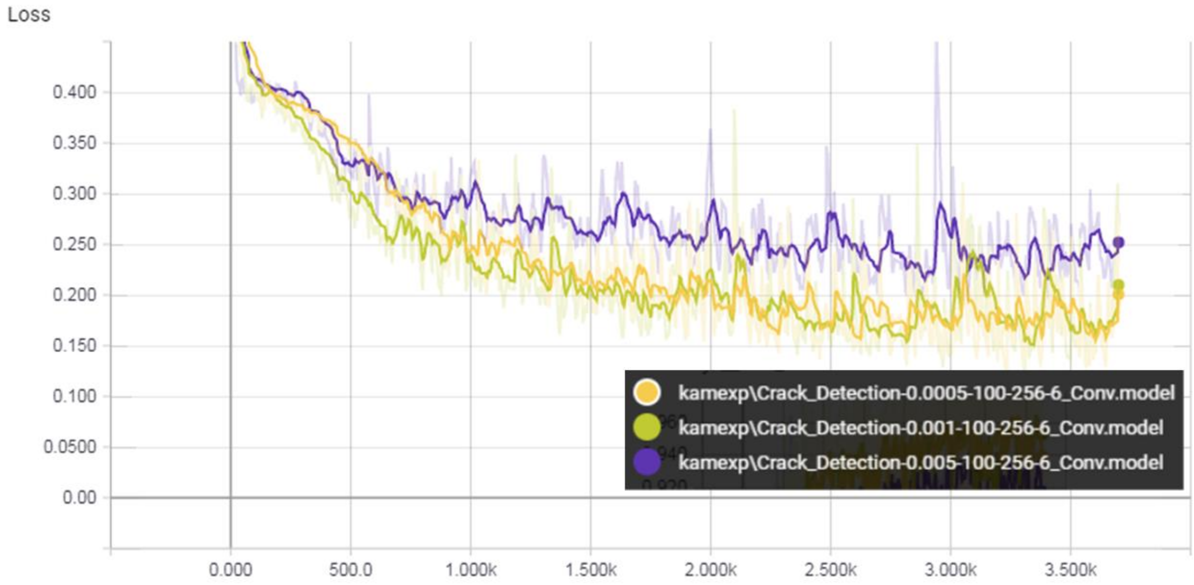


Figure 18: Effect of learning rate on the loss

5.3.1.3 Effect of dropout

The human brain is resilient to damage because it has redundancy and ability to create new neural connections. Dropout is a technique to simulate this process in neural networks by subsequently cutting neural network connections between layers at training time.

Dropout is a very efficient way to reduce overfitting of the training data. The dropout percentage α for the fully connected layer was varied between 0.3 and 0.7. The effect on the accuracy and the loss is shown in Figure 19 and Figure 20: the results show that dropout percentage α, does not have an effect on the accuracy or the loss globally. However the smallest value of α resulted in less local fluctuation in the loss curve.

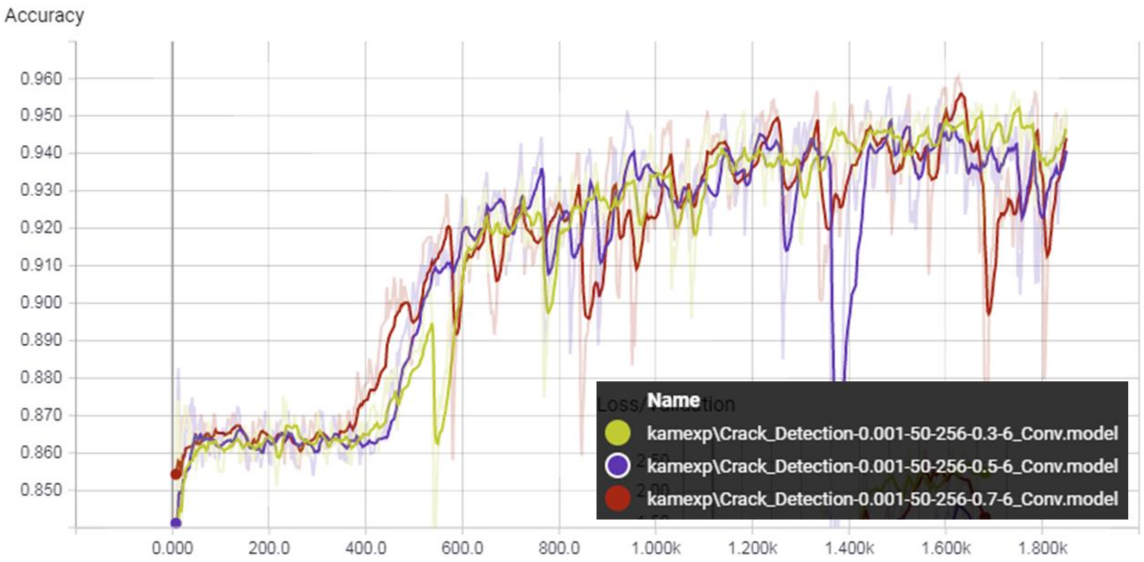


Figure 19: Effect of dropout on the accuracy

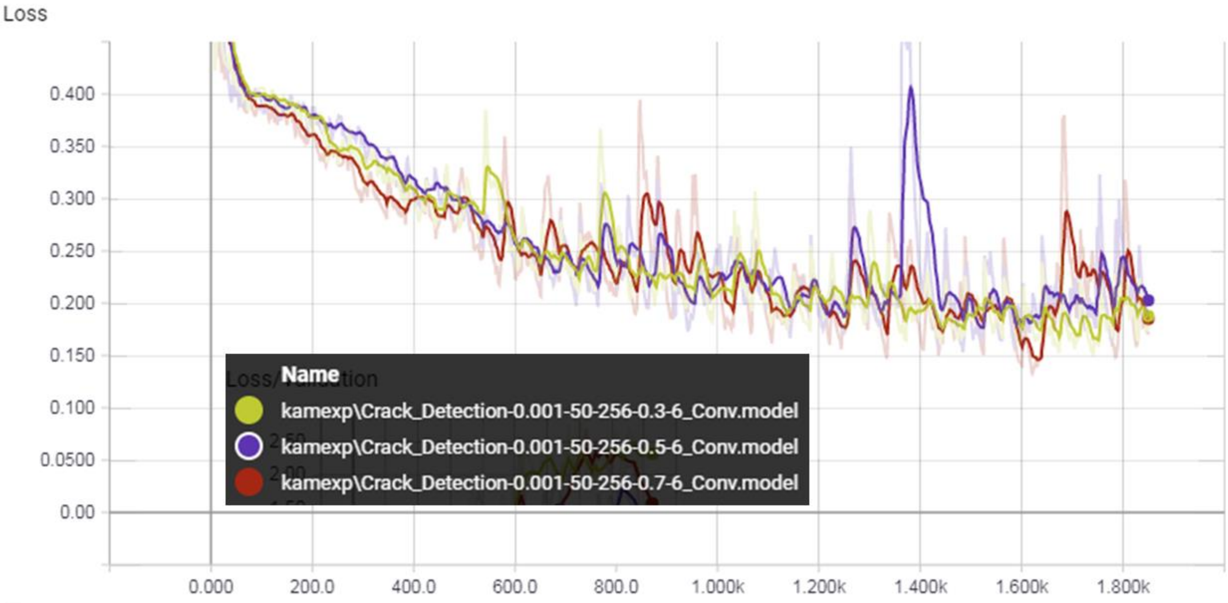


Figure 20: Effect of dropout on the loss

5.3.2 Prediction and results

5.3.2.1 Effect of filter sizes

A filter is a matrix used by the convolution layer to multiply over different sub images of the image and assign scores in order to detect variations.

In order to investigate the effect of filter size, the matrix size was varied and its effect on accuracy was checked.

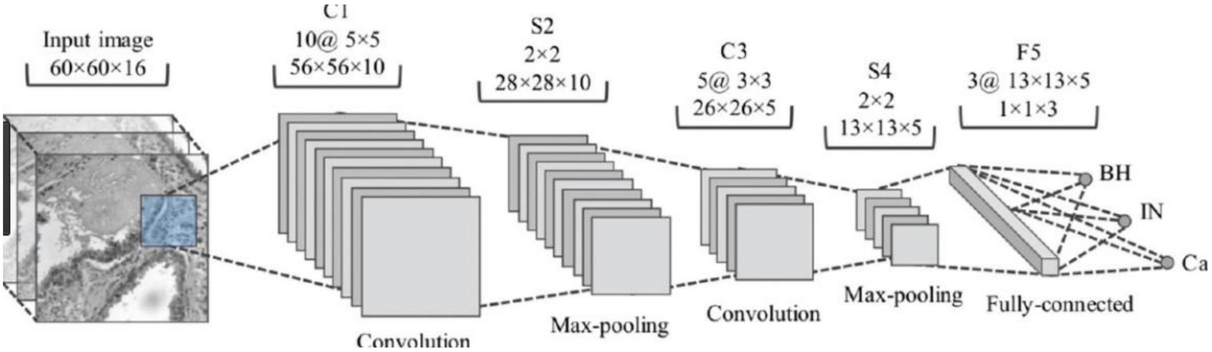


Figure 21: General CNN Architecture demonstrating the functioning of a convolutional architecture showing different sizes of filters (Hassan et all, 1970)

The effect of filter size on classification accuracy cannot be generalized for all problems and works differently for each kind of classification problem. This experiment was designed to study the effect of filter size on the road crack classification problem and the result of filter size on classification accuracy can be seen in Figures 22 and 23.

In the implemented test runs presented in Figures 22 and 23, the orange line indicates 5x5 filter size for a 3 layer convolutional network and the purple line indicates a 3x3 filter size for a 3 layer convolutional network. It can be seen that both training and validation accuracies are better for the 5x5 filter size indicating that it was more suited to the problem.

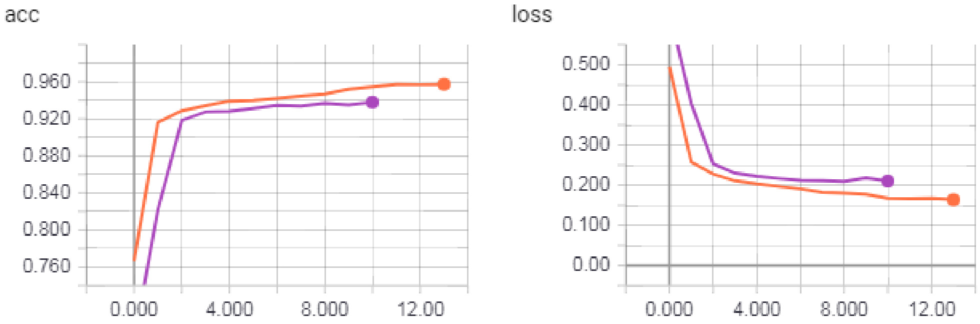


Figure 22: Training accuracy and loss vs number of epochs (Orange line 5x5 filters and Purple line 3x3 filters)

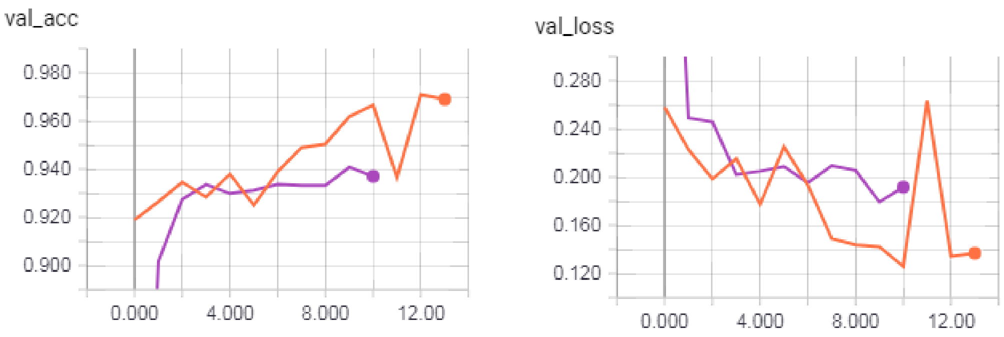


Figure 23: Validation accuracy and loss vs number of epochs (Orange line 5x5 filters and Purple line 3x3 filters)

5.3.2.2 *Effect of number of layers*

As the number of layers increases in a CNN architecture its capability to characterize complex shapes (like facial expressions and features etc.) increases. The downside to increasing the number of layers was that they are harder to train and need more data since the number of parameters increases. This requires significant computing power which may not always be available.

5.3.2.3 *Prediction results*

The output of the convolution operation at layer number 3 can be visualized and is observed to be as follows.

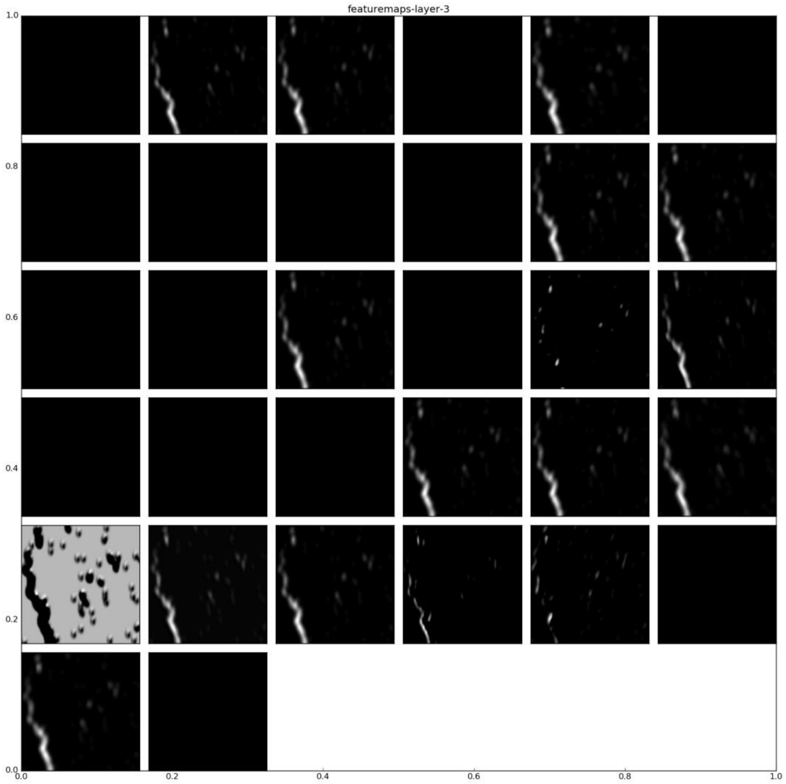


Figure 24: Image seen through the third CNN layer

It can be observed that the classifier was able to train on crack detection and was able to identify the important regions as crack regions.

The confusion matrix obtained was as follows. The matrix indicates on the y-axis the number of images with originally given labels and the x-axis indicates predicted labels. It can be seen that around 90% of the images were on the diagonal which indicated they were classified correctly.

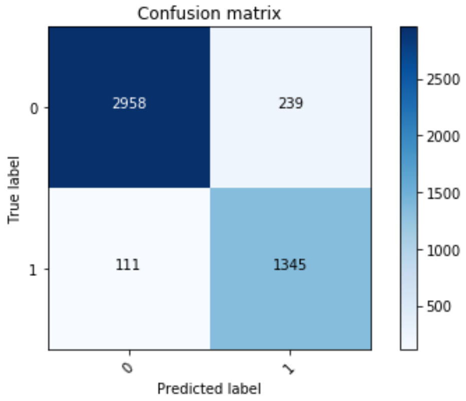


Figure 25: Confusion matrix for a 5x5 filter size CNN

5.3.2.4 *Effect of adding Spatial Transform Network (STN)*

The addition of Spatial Transform Networks was explored and the filters learned in case of spatial transform nets were visualized in Figure 26 to understand the training process. It was found that the learned filters looked like gabor filters (Medina et all, 1970).

Also it was found that some of the filters had specific shapes which might signify other filters were also important and were learned in addition to the normal gabor filters.

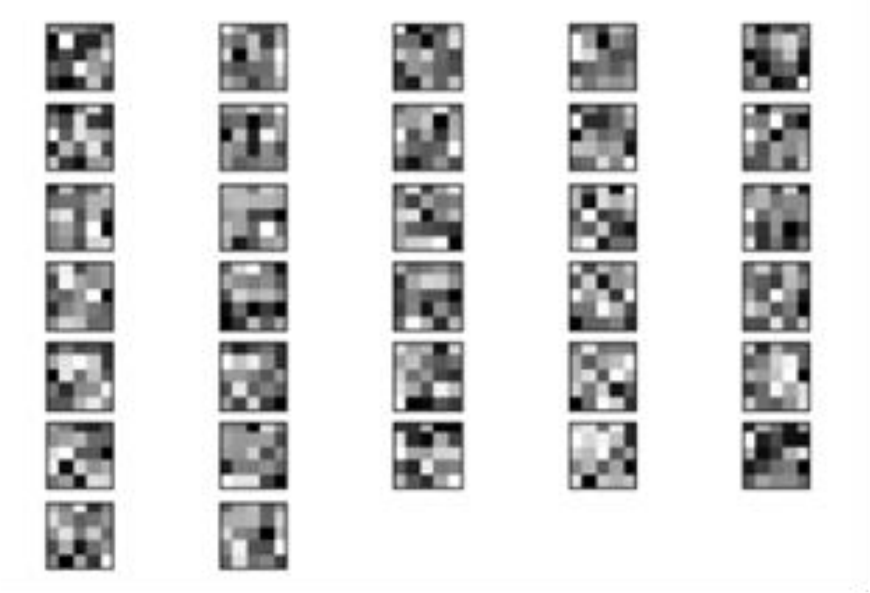


Figure 26: Visualizing weights in layers

It was observed that adding a spatial transform network led to the reduction of validation accuracy from 95 % to 84 %.

5.4 Conclusions and recommendations

5.4.1 Conclusions

The parametric investigation showed that:

- A better accuracy can be achieved by increasing the batch size; the increase of batch size also has an effect on reducing the local variation in the accuracy curve and the loss curve.
- Decreasing the learning from 0.005 to 0.001 yielded an increase of 2.4% in the accuracy; a further decrease to 0.0005 did not make much difference in terms of overall accuracy.
- Increasing the dropouts for the fully connected layer did not make a difference in the overall accuracy.
- Increasing the filter size from 3x3 to 5x5 had the effect of increasing slightly the accuracy in our data set. Using a 3 convolutional-layer neural network with a 5x5 filter, a binary classification to separate cracked from un-cracked yielded an accuracy of 92.5% of the images being correctly classified as either cracked or un-cracked. The proportion of images with a crack that were incorrectly classified as not cracked was 2.4%.
- The use of a larger dataset and manually separating images with good distinguishable crack features resulted in an increase in the test accuracy from 92.5 % to 95%.
- Inclusion of spatial transform networks reduced the test accuracy from 95 % to 84 % and hence was not suitable to the problem.
- Training of AlexNet for this problem resulted in inadequate training because the CNN architecture had too many parameters and the size of the dataset was inadequate to learn them all. A larger dataset could produce a better result in this situation.

5.4.2 Further work

Initial thoughts on further work:

1. Use Laser Crack Measurement System (LCMS) data sets for which there are reference data sets (the data set could be from a manual analysis or an output of algorithms with known performance). This data set combines information from images and 3D profile to get better predictions.

2. Use of forward facing images, retro-reflectivity and Light Detection and Ranging (LIDAR) data to detect changes in non-pavement assets; for example information

panels that come in different shapes and sizes to assess their condition in terms of alignment, retro properties etc.

3. Detection of drains and manholes with a convolutional neural network. For this, accurate data for the position of drains and manholes is available.
4. Existing algorithms have difficulty in detecting give-way signs and slow signs using retro-reflectivity data. To overcome this, the idea is to combine forward facing video synchronised with retro-reflectivity and feed this into CNN in order to find out its location and identify where a road sign correlates with retro-reflectivity.
5. Investigate applications with satellite images for example to detect features that look like roads, rivers etc.

6 Discussion

Overall, the studies show that machine learning can be applied to numerous areas of the transport industry to help analyse known problems and build frameworks for future research. No one method fits all problem types so careful initial analysis of individual problems is needed to identify the most suitable approach. The transport industry already has a vast array of data sets waiting to be explored and exploited, some of them could even be linked together to enrich the information contained within. Even in those areas where data is scarce it is still possible to identify trends and recommend how to proceed further.

7 Overall Conclusions

Three very different case studies have been presented here using different techniques to achieve different aims and different results.

7.1 Case study 1

The train driver behaviour study successfully used clustering to analyse a small data set and still provide some useful conclusions. In the end, it was possible to conclude that a machine learning approach would be able to hint at correlations similar to those recognised by a human, but making it approach manual observation would require more data. This case study highlights how important the data structure is in machine learning in order to produce worthwhile results.

7.2 Case study 2

The project has resulted in a stable data retrieval system that made it possible to quickly query the database to collate any number of inputs and outputs from those available and produce a single global model for the dataset or a multitude of section-specific ones. The script also allowed for the assessment of individual models, and could perform a combined aggregate test on several trees at once. The study showed that the resources of existing data sets is significant. The final results were inconclusive but the act of researching this data has now built a stable framework on which to progress future work.

7.3 Case study 3

The crack detection study showed that some of the more mundane and labour intensive processes can be automated and useful results obtained.

The investigation showed that the outcome can vary quite significantly by varying parameters. Some experiments lead to an improvement in the accuracy of crack detection, whilst other reduced it. The experience gained with these experiments has helped gain an increase in knowledge on how machine learning can be applied to a problem.

8 Summary

These case studies have highlighted that machine learning can be used across the whole transport industry but the results will vary vastly depending on the problem being tackled. The train driver behaviour study successfully used clustering to analyse a small data set and still provide some useful conclusions. This study also emphasised the need to fully pre-process what little data they had to avoid outlier points from obscuring any visible trends.

The condition forecasting of road pavements study shows that there are vast resources of existing data sets out there waiting to be explored. The final results were inconclusive but the act of researching this data has now built a stable framework on which to progress future work. This study also offers the possibility of extending a known data set by linking in other data resources.

The crack detection study using classification of images shows that we can automate some of the more mundane and labour intensive processes. The methodology used here required a lot of tweaking to find the right parameter values to use but once set they produced some useful results.

Overall, these studies show that machine learning can be applied to numerous areas of the transport industry to help analyse known problems and build frameworks for future research. No one method fits all problem types so careful initial analysis of individual problems is needed to identify the most suitable approach to use. The transport industry already has a vast array of data sets waiting to be explored and exploited, some of them could even be linked together to enrich the information contained within. Even in those areas where data is scarce it is still possible to identify trends and recommend how to proceed further.

9 Recommendations

The work described here covers three separate case studies that show how machine learning could be applied. The work needs to continue in order to gain more experience with the methods employed and what methods suit the various different problems. As the experience grows the understanding of how machine learning can be applied will grow and an increasing number of problems can be solved. As time passes, machine learning will become further relied upon to deal with large amounts of data, finding new ways to address issues in more efficient and effective ways.

The outlook for the future of ML is bright. It is becoming increasing important due to both the availability of greater amounts of data and the increasing computing power that permits better analysis. There remains the need to learn more about the potential and how to harness it in terms of making analysis more effective and enabling handling of larger and more diverse data sets.

10 Acknowledgements

We would like to thank Highways England for granting permission to use network, condition and construction data from Highways England’s Pavement Management System (HAPMS).

11 References

Haralick R.M., Shanmugam K. and Dinstein I. (1973), Texture features for image classification. IEEE Transaction on Systems, Man and Cybernitics, Vol. 3 No 6. pages 610-621.

Hassan H., Chaddad A., Harkouss Y., Desrosiers C., Toews M., Tanougast C. (1970). Classifications of multispectral colorectal cancer tissues using convolution neural network. Journal of Pathological Information, vol.8:1.

Laws K. (1980). Textured Image Segmentation. Ph.D. Dissertation, University of Southern California.

LeCun Y., Bottou L., Bengio Y. and Haffner P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, vol. 86, pages 2278–2324.

CSE576, 2000. Computer Vision: Chapter 7 (PDF), pages 9-10, University of Washington.

Jensen H., Legakis J. and Dorsey J. (1999). Rendering of wet materials. Rendering techniques 99. Editions: Lischinski, D. and Larson, G. Springer-Verlag, pages 273-282.

Introduction to Boosted Trees (2015). Retrieved 12/1/2017 from XGBoost: http://xgboost.readthedocs.io/en/latest/model.html

Kuehnle A. and Burghout W. (1998). Winter road condition recognition using video image classification. Transportation research record 1627, pages: 29-33.

LeCun Y., Boser B., Denker J.S., Henderson D., Howard R.E., Hubbard W. and Jackel, L.D. (1989). Backpropagation applied to handwritten zip code recognition. Neural Computation. No. 1, Vol. 4, pages 541–551.

Ojala T. (1996). A comparative study of texture measures with classification based on feature distributions. Pattern recognition, Elsevier science Ltd.

Ojala T., Pietikäinen M. and Harwood D. (1996). A comparative study of texture measures with classification based on feature distributions. Pattern Recognition vol. 29, pages 51-59.

Ojala T. and Pietikainen M. (1999). Unsupervised texture segmentation using feature distributions. Pattern recognition, vol. 32, pages: 477-486.

McFall K. (2000). Artificial neural network technologies applied to road condition classification using acoustic signals. In 10th standing international road weather congress, pages 189-204.

Medina R., Llamas J., Gómez-García-Bermejo J., Zalama Z., Segarra M.J. (1970). Crack detection in concrete tunnels using a gabor filter. Sensors, vol.17, 1670.

Pietikainen M., Ojala T. and Xu Z. (2000). Rotation-Invariant classification using feature distribution. Pattern recognition, vol. 33, No. 1, Elsevier, pages 43-52.

Jarrett K., Kavukcuoglu K. and LeCun Y. (2009). What is the best multistage architecture for object recognition? In Proceedings of International Conference on Computer Vision, pages 2146–2153.

Teshima T., Saito H., Shimizu M. and Taguchi A. (2009). Classification of wet/dry area based on the Mahalanobis distance of feature from time space image analysis. IAPR conference on machine vision applications, pages 467-470.

Kavukcuoglu K., Sermanet P., Boureau Y., Gregor K., Mathieu M. and LeCun Y. (2010). Learning convolutional feature hierarchies for visual recognition. In Proceedings of Neural Information Processing Systems (NIPS).

Xu P., Yao, H. Ji, R., Sun X. and Liu X. (2010). A rotation and scale invariant texture description approach. Visual communication and image processing 2010, Proc. of SPIE7744, vol. 7744.

Zeiler M., Taylor G. and Fergus R. (2011). Adaptive deconvolutional networks for mid and high level feature learning. In Proceedings of International Conference on Computer vision (ICCV).

Krizhevsky A., Sutskever I. and Hinton G. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In Proceedings of Neural Information Processing Systems (NIPS).

Hinton G.E., Srivastava N. , Krizhevsky A., Sutskever I., Salakhutdinov R.R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arxiv.1207.

Coates A., Huval B., Wang T., Wu D., Ng A. and Catanzaro B. (2013). Deep learning with COTS HPC. In Proceedings of International Conference on Machine Learning (ICML).

Kim Y., Baik N. and Kim J. (2013). A study on development of mobile road surface Condition Detection System Utilizing Probe Car 1. Journal of emerging trends in computing and information sciences.

Yacoob Y. (2013). Matching dry to wet material. IEEE international conference on computer vision, pages 2952-2959.

Zeiler M. and Fergus R. (2013). Visualizing and understanding convolutional networks. arXiv:1311.2901, pages 1–11.

Kingma D.; Ba, Jimmy B. (2014). Adam: A method for stochastic optimization. arxiv.1412.

Yang H., Jang H., Kang J. and Jeong D. (2014). Classification algorithm for road surface condition. IJCSNS International journal of computer science and network security, vol. 14, No. 1.

Appendix A Literature

In this section we will cover two main streams of machine learning for image processing, the classical machine learning which is the main approach taken by researcher pre 2010, and the modern machine learning, that focuses on deep learning technologies such as Convolutional Neural Network, De-Convolved Neural Network and Recurrent Neural network. New developments in computer processing technologies have contributed tremendously to the taking off of deep learning.

This section starts by covering the classical machine learning, then the modern machine learning known also as deep learning.

From this point onward any mention of the word “learning” is under the “supervised learning” context.

A.1 Classical machine learning

In any machine learning approach (be it classical or modern) there are two phases: the training phase and the prediction phase.

The two main ingredients of the training phase are feature extractions, and training of the machine learning algorithm to classify the images. Basically the raw images transformed into a set of vectorised features and the labels are input to the machine learning algorithm that produces a trained classifier that will be used in the prediction phase.

In the prediction phase, raw image transformed into a set of vectorised features as an input to the trained classifier that outputs class labels to the raw images.

In the following a review of some classical feature extractions are given in section A.1, some popular machine algorithms in section A.3, and a methodology for building a training data set is described in A.4.

A.2 Review of Feature extraction

Feature extraction is the backbone of the classical machine learning. Methods for feature extraction using images are based on computing a set of statistics from the distributions of local features in the neighbourhood of the pixel. In the following some techniques popular among researchers are described.

A.2.1 Co-occurrence matrix

The co-occurrence method was introduced by Haralick et al. (1973) as a measure of the texture of an image. A co-occurrence matrix is a matrix that is defined over an image to be the distribution of co-occurring pixel values at a given offset d in any direction θ. Mathematically a co-occurrence matrix C is defined over an image I of size n×m for a displacements Δx and Δy as:

$$C(i,j,d,\theta)=\sum_{p=1}^{n}\sum_{q=1}^{m}\begin{cases}1, & \text{if} \quad \mathrm{I}(\mathrm{p},\mathrm{q})=i \text{ and } \mathrm{I}(\mathrm{p}+\Delta \mathrm{x}, \mathrm{q}+\Delta \mathrm{y})=j\\ 0, & \text{otherwise}\end{cases}$$
$$\text{with } d=\sqrt{\Delta \mathrm{x}^2+\Delta \mathrm{y}^2} \text{ and } \theta = a\tan\left(\frac{\Delta \mathrm{y}}{\Delta \mathrm{x}}\right) \quad \textbf{(A 1)}$$

The Co-occurrence matrix C is often referred to as Grey Level Co-occurrence Matrix (GLCM). Usually the matrix C is calculated for values of θ show in Figure A 1 depicts an eight neighbourhood pixel to describe the connectivity to the central pixel (shown by the black dot): pixels 1 and 5 are 0^o neighbour to the central pixel, pixels 2 and 6 are 135^o neighbour to the central pixel, pixel 7 and 3 are 90^o neighbour to the central pixel; and pixels 8 and are 45^o neighbour to the central pixel. Figure A 1 is the case for which d is equal to 1, however the value of d will depend on the application .i.e. if we want to quantify fine texture or coarse texture; there is no recommended value for d in the literature, Haralick et al used d equal to 1.

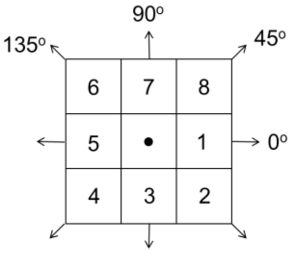


Figure A 1: Eight neighbourhood pixel (1, 2, 3, 4, 5, 6, 7 , and 8) connectivity to the central pixel (black dot)

This matrix is used to extract second-order statistical texture features. Haralick et al. suggested 14 features to describe the two dimensional probability density function p(i,j) defining the (i,j) entry after normalising GLCM. Among them are angular second moment, contrast, correlation, entropy, sum of entropy, information measure of correlation and the maximal correlation features defined respectively as:

Angular second moment:

$$f_1=\sum_i\sum_j\{p(i,j)\}^2 \quad \textbf{(A 2)}$$

Contrast:

$$f_2=\sum_{n=0}^{N-1}n^2\left\{\sum_{i=1}^{N}\sum_{j=1}^{N}p(i,j)\right\} \qquad \text{with} \quad \mathrm{n}=|\mathrm{i}-\mathrm{j}| \quad \textbf{(A 3)}$$

Correlation:

$$f_3=\frac{\sum_i\sum_j ijp(i,j)-\mu_x\mu_y}{\sigma_x\sigma_y} \quad \textbf{(A 4)}$$

where μ_x, μ_y, σ_x and σ_y are the means and the standard deviations of the marginal probability matrix p_x (obtained by summing the rows of p(I,j)) and p_y (obtained by summing the columns of p(I,j)).

Entropy:

$$f_9 = -\sum_i \sum_j p(i,j) log(p(i,j)) \quad \textbf{(A 5)}$$

Sum Entropy:

$$f_8 = -\sum_{i=2}^{2N} p_{x+y}(i) log\{p_{x+y}(i)\} \quad \textbf{(A 6)}$$

where $p_{x+y}(i)$ is defined as:

$$p_{x+y}(k) = \sum_{i=1}^{N} \sum_{\substack{j=1 \\ i+j=k}}^{N} p(i,j) \quad \textbf{(A 7)}$$

Information measures of correlation:

$$f_{12} = \frac{HXY - HXY1}{max(HX, HY)} \quad \textbf{(A 8)}$$

$$f_{13} = (1 - exp[-2(HXY2 - HXY)])^{1/2} \quad \textbf{(A 9)}$$

where *HX* and *HY* are entropy of the marginal probabilities p_x and p_y obtained by summing the rows and the columns of the normalised grey tone spatial dependence matrix *p*(*i,j*).

$$HXY = -\sum_i \sum_j p(i,j) log(p(i,j)) \quad \textbf{(A 10)}$$

$$HXY1 = -\sum_i \sum_j p(i,j) log\{p_x(i) p_y(j)\} \quad \textbf{(A 11)}$$

$$HXY2 = -\sum_i \sum_j p_x(i) p_y(j) log\{p_x(i) p_y(j)\} \quad \textbf{(A 12)}$$

Maximal correlation coefficient:

$$f_{14} = (\text{Second larges eigen valu e of } Q)^{1/2} \quad \textbf{(A 13)}$$

where

$$Q = \sum_k \frac{p(i,k) p(j,k)}{p_x(i) p_y(j)} \quad \textbf{(A 14)}$$

A.2.2 Difference method

It is the same as co-occurrence matrix, in the sense it uses displacements Δx and Δy to locate a pixel from position (i,j) in an image I, and then compute the grey scale level absolute difference as:

$$I'(i,j) = |I(i,j) - I(i + \Delta x, j + \Delta y)|$$

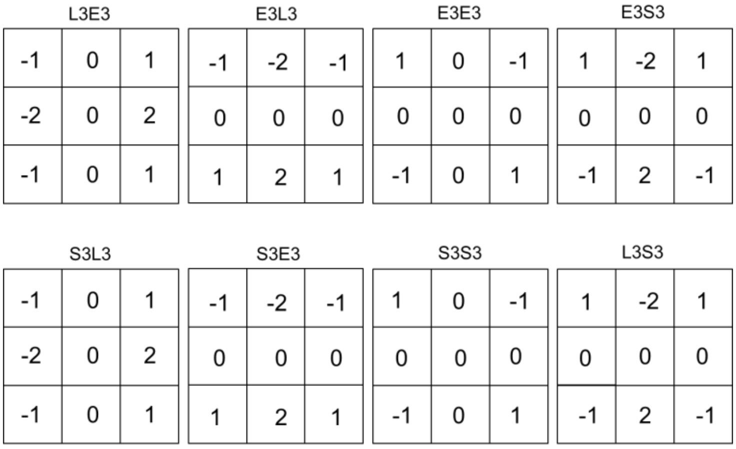
Let p’ be the probability density function of I’(x,y). If the image has got m grey scale levels; then p’ is a vector of size m, and its ith component is the probability that I’(i.j) is equal to i.

The difference method was used by Ojala et al. (1994) and computed four texture features: DIFFX and DIFFY for probability density of grey scale difference between neighbouring pixels in the horizontal and vertical directions; DIFF2 accumulates absolute difference in the horizontal and vertical direction, and DIFF4 in all four principal directions.

A.2.3 *The Law method*

The Law method (Law, 1980) uses local masks to detect various types of textures. The masks are built from 3 elements vectors or 5 elements vectors.

The three elements vectors are L3=[1,2,1] used for local averaging , E3=[-1,0,1] for edge detection and S3=[-1,2,-1] for spot detection and are used to generate Law's 3 by 3 masks shown in Figure A 2.



L3E3			E3L3			E3E3			E3S3		
-1	0	1	-1	-2	-1	1	0	-1	1	-2	1
-2	0	2	0	0	0	0	0	0	0	0	0
-1	0	1	1	2	1	-1	0	1	-1	2	-1

S3L3			S3E3			S3S3			L3S3		
-1	0	1	-1	-2	-1	1	0	-1	1	-2	1
-2	0	2	0	0	0	0	0	0	0	0	0
-1	0	1	1	2	1	-1	0	1	-1	2	-1

Figure A 2: Law's 3×3 masks

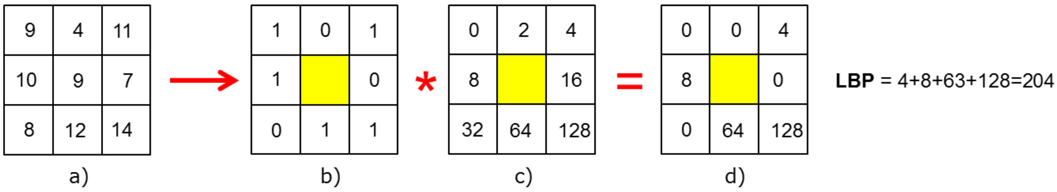
The masks are five element vectors and are given below: L5 is used to quantify the local average of the texture; E5 is used to extract the edges; S5 is a spot detector; W5 is a wave detector; and R5 is a ripple detector

L5=[1,4,6,4,1], E5=[-1,-2,0,2,1], S5=[-1,0,2,-1], W5=[-1,2,0,-2,1], R5=[1,-4,6,-4,1]

These vectors are used to generate sixteen 5×5 convolution mask used to calculate the energy of texture which is then represented by a nine element vector for each pixel, see CSE576, 2000. The list of masks is: L5L5, E5E5, S5S5, R5R5, L5E5, E5L5, L5S5, S5L5, L5R5, R5L5, E5S5, S5E5, E5R5, R5E5, S5R5, R5S5.

A.2.4 *Local binary pattern*

Ojala et al. (1996) introduced the Local Binary Pattern (LBP) operator, it provides a robust way of describing locally pure binary patterns in a texture. It is invariant to any monotonic grey scale transformation; Figure A 3 depicts the method for calculating LBP.



9	4	11
10	9	7
8	12	14

a)

1	0	1
1		0
0	1	1

b)

0	2	4
8		16
32	64	128

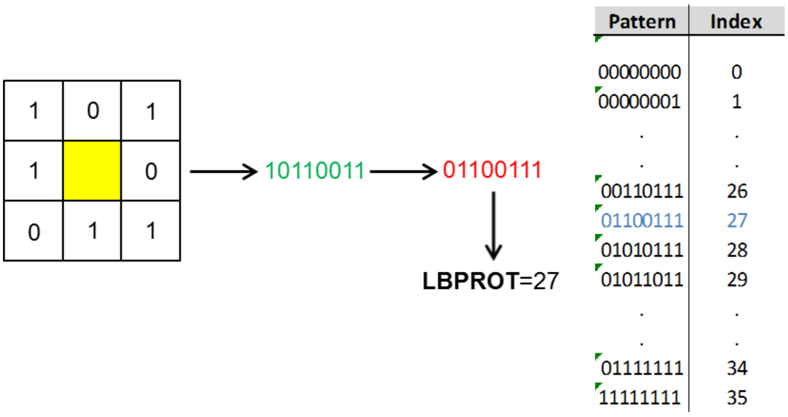
c)

0	0	4
8		0
0	64	128

d)

Figure A 3: Calculation of LBP

Ojala also introduced a variant of LPB that is rotation invariant LPBROT. The binary values of the thresholded pattern in **Error! Reference source not found.** are mapped into 8bit words highlighted green in **Error! Reference source not found.**. Then binary shifts are applied until the 8bit word matches on of the 36 pattern displayed in Figure A 4. Observe that after one shift a match is obtained for pattern with index 27 used as the value for LBPROT.



1	0	1
1		0
0	1	1

Pattern	Index
00000000	0
00000001	1
.	.
.	.
00110111	26
01100111	27
01010111	28
01011011	29
.	.
.	.
01111111	34
11111111	35

Figure A 4: Calculation of LBPROT

A.2.5 Wavelet transform

Wavelet transform is a mathematical entity that provides a mapping of a function from the time domain to frequency/time domain. Wavelet transforms are different from Fourier transforms, firstly wavelet are local and the Fourier transforms are global, secondly wavelet transforms provide a good resolution in both the time and the frequency domain whereas Fourier transforms provide a good resolution only in the frequency domain. Wavelet transforms have been used to obtain a robust characterisation of texture of the road surface, for example Xu et al. (2010) presented a novel texture description approach that is robust to variances in rotation, scale and illumination; to achieve this they used the Local Haar Binary Pattern (LHBP) with a feature extraction and scale self-adaptive classification. They combined the LHBP feature constructor with a threshold filter to remove the variances of grey level caused by changes of light.

Yang et al. (2014) used wavelet transform to characterise texture from images obtained with a stereo camera-based mobile image processing system to detect road surface condition: Dry, wet, snowy and icy. They used a combination of feature construction, wavelet statistical features and Hue intensity histograms, to improve the accuracy of the classification.

A.3 Machine learning methods

Many papers for road condition classification were identified using classification methods such as support vector machine (Yang et al., 2014; Shu et al., 2007; Xu. et al., 2010), k-mean clustering (Kim et al., 2013), nearest neighbour algorithm (Teshima et al., 2009; Ojala, 1996), neural networks (Kuehnle et al. 1998, McFall, 2000). One paper used G statistics to classify natural scenes with water (Ojala et al., 1999).

A.3.1 G statistics

Most of the approaches to texture classification quantify texture measures by single values (means, variances etc.), which are then concatenated into a feature vector. In this way, much of the important information contained in the whole distributions of feature values is lost, Pietikainen et al. (2000).

They used the G statistics which a log-likelihood pseudo metric to compare feature distribution during classification. The value of the G statistic indicates the probability that the two sample distributions come from the same population, the higher the value, the lower the probability that the two samples are from the same population. The G statistics is expressed with the following formula:

$$G = 2\sum_{i=1}^{n} s_i \log \frac{s_i}{m_i} \tag{A 15}$$

Where s and m are sample and model distribution, n is the number of bins and s_i, m_i are the sample probability and the model probability at bin i, respectively.

A texture class is represented by a number of model samples that are ordered according to the probability that they are coming from the same distribution as the test sample being classified. This probability is measured by a two-way test of interaction as:

$$G = 2\left(\begin{aligned} &\sum_{s,m}\sum_{i=1}^{n} f_i \log f_i - \sum_{s,m}\left(\sum_{i=1}^{n} f_i\right)\log\left(\sum_{i=1}^{n} f_i\right) - \\ &\sum_{i=1}^{n}\left(\sum_{s,m} f_i\right)\log\left(\sum_{s,m} f_i\right) + \left(\sum_{s,m}\sum_{i=1}^{n} f_i\right)\log\left(\sum_{s,m}\sum_{i=1}^{n} f_i\right) \end{aligned} \right) \tag{A 16}$$

Where s, m are the two sample histograms, n is the number of bins and f_i is the frequency at bin i. The more alike the histograms s and m are, the smaller is the value of G.

A.3.2 Support vector machine

The Support Vector Machine (SVM) is a technique that is used to classify the space of feature using a surface. Support Vector Machine (SVM) is a supervised learning algorithm that analyse data and recognises patterns. Given a set of training samples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new samples into one class or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the training samples as points in space, mapped so that the samples of the separate classes are divided by a clear gap that is as wide as possible. New samples are then mapped into that same space and predicted to belong to a class depending on which side of the gap they fall on. In addition to performing linear classification, SVM can efficiently perform a non-linear classification using non-linear kernels, implicitly mapping their inputs into high-dimensional feature spaces.

A.3.3 Neural networks

Neural Nets can be considered as a general purpose fitting algorithm as they are able to fit complex nonlinear model. The general structure of a neural network is to mimic the neuron linkage and transmission in the brain. The first layer of the neural network is the input nodes representing the data points, this layer sends data via synapse (representing weights) to a hidden layer representing the neuron; and depending on the complexity of the neural system, the hidden layer could propagate data via synapse linking several hidden layers, then finish at the final layer which is the output nodes.

The main parameters of the Neural Networks are:

- N observation points
- The interconnection defining the different layers
- The weight representing the synapse, and the process of updating these parameters.
- The activation functions that represent the transformation that occur in the neural nodes; this tries to mimic the firing of the brain synapses

A.3.4 Decision trees

Decision tree is a very powerful technique that is used for regression or classification; we speak of regression if the data sets analysed are continuous, and classification if they are discrete. Continuity of data set implies the use of real numbers that can be described with continuous function. Texture data collected by a laser scanner are an example of continuous data. Discrete implies the use of categorical data (that are basically descriptive or quality attributes of the data set) for example accident count on the road, and the level of deterioration of a road classified as high, medium or low. Basically the decision tree could be viewed as a process of organizing the interaction and decision outcome of several input variables in a controlled systematic way.

A.3.5 k-mean clustering

k-mean clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space similar to Voronoi cells.

A.3.6 Gaussian mixtures

They are type of algorithm know as unsupervised learning, in the sense that there is no prior knowledge on how the data set relates to the mixture models; and the knowledge about this relation to the data set is acquired during the successive iterations trying to fit the mixture models to the data set. This gives generality to the Gaussian model and makes it able to detect complex relationship between variables.

The main parameters of the model are:

- N observation points
- K Gaussian probabilities
- A set of probabilities that pin a particular data point to a particular Gaussian distribution.
- The mean the Gaussian probability
- The variance of the Gaussian probability

The last three parameters are updated during the iteration process in order to updated the count of data points falling in each Gaussian, the mean of the Gaussian which has physical meaning that is position of the Gaussian in the space of variable and the covariance expresses the spread of the Gaussian in the variable space.

In the following I will try to explain the methodology to use to perform a classification with images, and the requirements for a good classification.

A.4 Methodology for building a training data set

The data set is split randomly into two halves, the first half is called the training set and the second half is called the out of bag set. The first set is used to train the classifier, and the second set is used to assess the predictive accuracy of the classifier.

Commonly, the training of the classifier uses supervised learning, in the sense that the training is performed on a data set for which the classification outcome is known.

In a classical machine learning context, the construction of this data set requires the definition of texture features using the techniques introduced in section A.1. A wide variety of images representing the different classes should be selected and labelled carefully, this task is very tedious and time consuming.

It should be stressed that adding data sources other than images will only make prediction more robust (this reinforces the view of using different sources of data in order to obtain accurate predictions); for example Kim et al. (2013) was able to detect wet road at a rate of 95% by combining road images and weather condition data; and Mc Fall (2000) reported that the classification of road conditions into the categories dry, wet, snowy and icy with Neural Networks is improved to 90% if additional measurements such as the recording of the sound of the a rolling tyre on the road surface are introduced in the classification.

An inconvenience of training a machine learning algorithm is that practically it is not possible to include all the possible cases of the features to perform complete training of different cases and variations, however with time and resources a more varied data set could be built, and the training of is updated regularly as new data is available. To do this we need to invest in efficient performing machine learning algorithms that do not take weeks to train.

A.5 Deep learning

The difference between machine learning and deep learning is that the feature extraction from the raw images is engineered by the data analyst to suit its application, this process is very time consuming and expansive for large and variable data; however in deep learning the feature extraction is a learning process (i.e. the feature are directly learning from the raw images), using a multitude of non-linear processing layers. The detail of these layers will be explained further for Convolutional Neural Networks (CNN) in section A.5.2.

A.5.1 A brief Historical overview of CNN

The first convolutional neural network was introduced in the early 1990's by LeCun et all (1989) in his seminal paper titled "Gradient-based learning applied to document recognition". In 1998, he proposed the CNN architecture (LeNet-5), given in Figure A 5 for digits recognition, it contains seven layers in total: two convolutional layers, two subsampling layers and three fully connected layers. It combines local receptive fields, weight sharing and spatial subsampling in order to ensure shift, scale and distortion invariance.

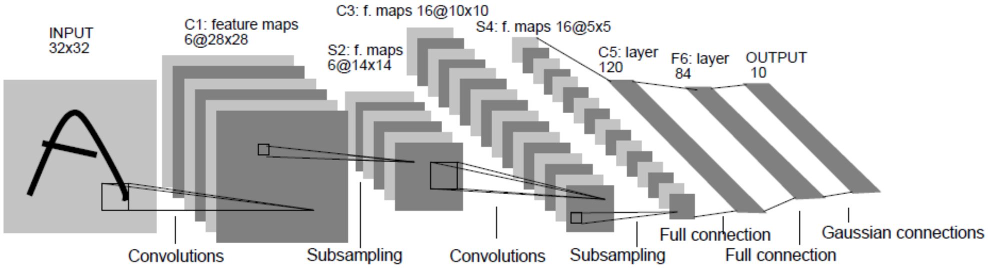


Figure A 5: CNN LeNet-5

Although CNN found advocates in image vision and image recognition pre 2012 (Jarret et all, 2009; Kavukcuoglu et all; 2010; Zeiler et all, 2013 and Coates et all, 2013); the use of CNN for image processing did not pick up only until 2012, after the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC), that achieved the lowest error score applied on real word image in the history of machine learning. The images used in ILSVRC are a subset of ImageNet and split into 1.4 million images with 1000 categories for training the CNN and 10000 images for predictions. Typical ImageNet categories used in the competition are shown in Figure A 6.

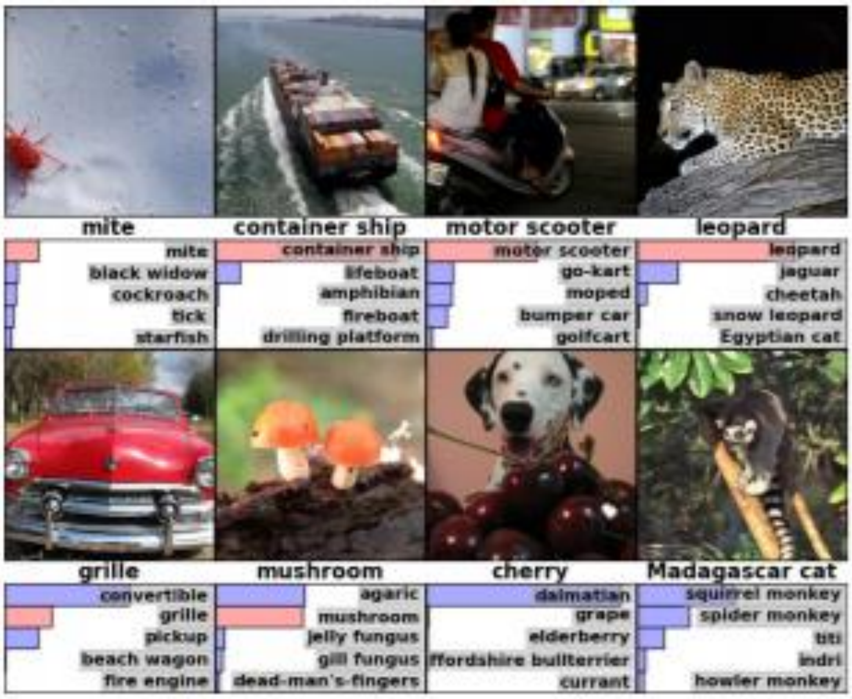


Figure A 6: Some ImageNet categories (after Krizhevsky et all, 2012)

In this competition, the AlexNet architecture achieved an error rate of 15.3% compared to 26% achieved by the second best. Figure A 7 depicts the AlexNet architecture, it is split between two GPU's (GPU are used instead of CPU's or in combination with them to increase the speed of calculations with parallel processing). AlexNet is made of five convolutional layers and three fully connected layers.

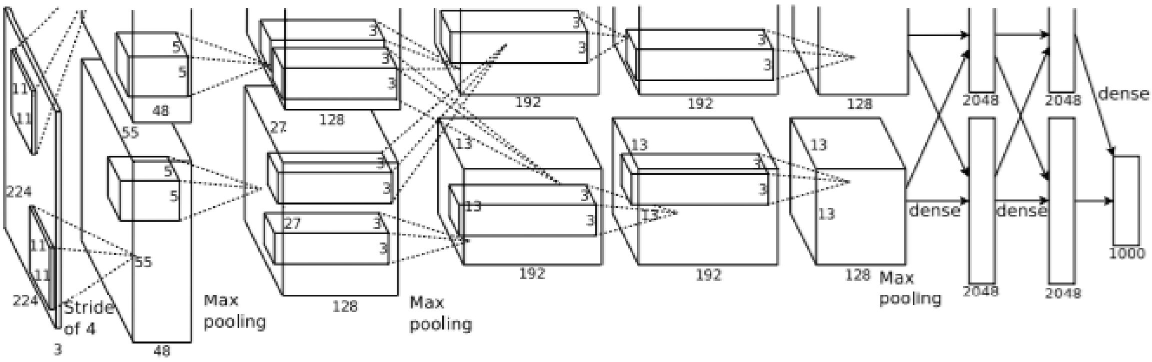


Figure A 7: Illustration of AlexNet architecture (after Krizhevsky et all, 2012)

The dramatic improvement in CNN performance is attributed (Zeiler et all, 2013) to the availability of much larger training data sets, with millions of labelled examples; powerful GPU implementations making the training of very large models practical and better regularisations strategies such as dropouts.

Zeiler et all (2013), proposed the De-Convolved Neural Network depicted in Figure A 8, basically it consist of a forward feed Convolutional Neural Network and reverse Convolutional Neural Net that has exactly the same characteristics as the forward feed CNN (in terms of filter sizes and number of layers) with the exception that Max Unpooling is performed instead of Max Pooling. The most important feature of the network is the Switches which provide a link between the forward and the reverse. The Switches are indexes for the location of the maximum activation response of the Max Pooling in the forward CNN.

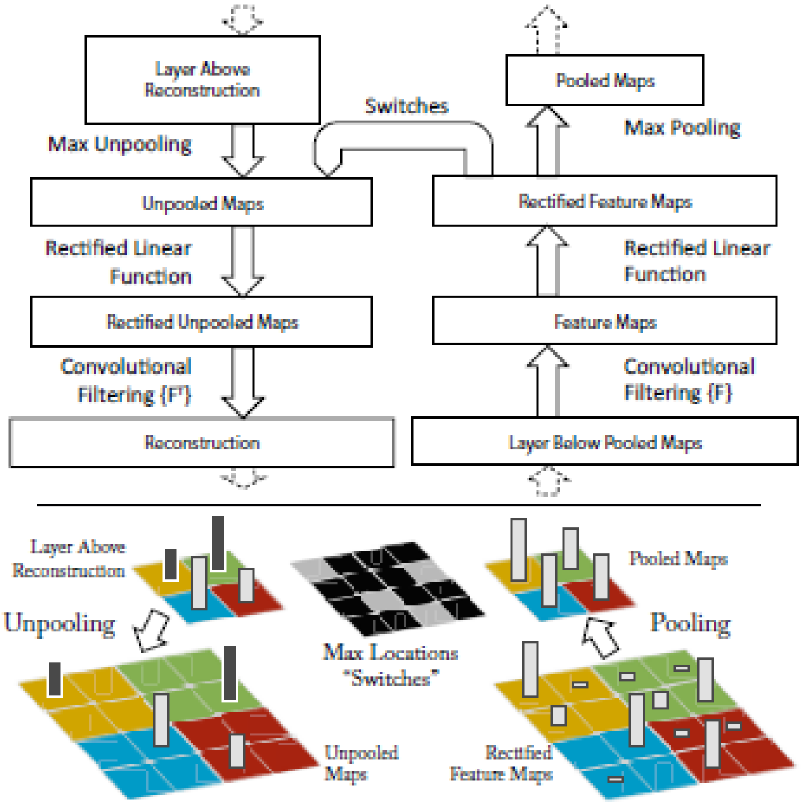


Figure A 8: Top: A DCNN layer (left) attached to a CNN layer (right). The DCNN will reconstruct an approximate version of the CNN features from the layer beneath. Bottom: An illustration of the unpooling operation in the DCNN, using switches which record the location of the local max in each pooling region (coloured zones) during pooling in the CNN. The black/white bars are negative/positive activations within the feature map (after Zeiler et all, 2013)

The DCNN was used as an assessment tool to investigate the shortcomings of the AlexNet, by doing this Zeiler et all won the ImageNet challenge by obtaining the lowest error rate of 11%.

A.5.2 A succinct explanation of CNN

A CNN is usually made of a number of convolutional layers, subsampling layers also known as Max Pooling layers and fully connected layers, see Figure A 9.

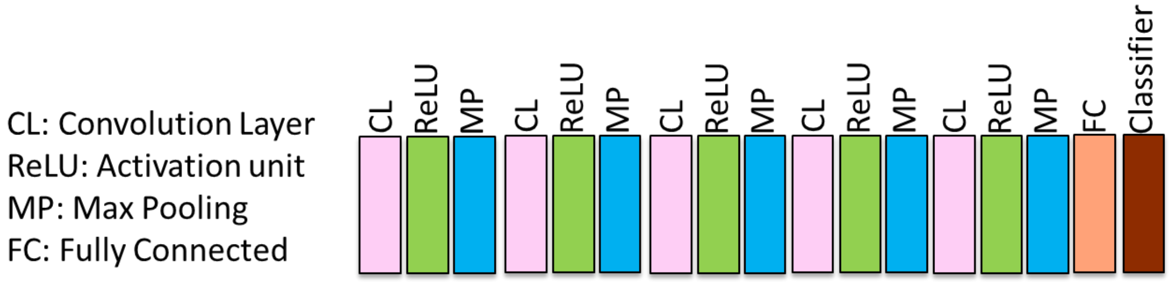


Figure A 9: A Typical Convolutional Neural Network (CNN) architecture

The convolutional layer, basically applies m filters to the image in order to learn semantic feature from it; the filter mask is generally square of size h, with a vertical or a horizontal translation degree of freedom with a step of s constrained to be between 1 and the size of the filter h. The filter transformation has two parameters that are learned: the weight w and the bias b. The convolution of the filters will produce m feature frames.

After each convolution a ReLu is applied to squash the data between 0 and 1, it similar to a sigmoid function or a Tanh function in what it is trying to achieve however with far better performance (see, Krizhevsky et all, 2012).

The Max Pooling layers are generally inserted after a convolution layer or after performing two or three successive convolutions as for the AlexNet architecture, see **Error! Reference source not found.**. The MaxPool layer has two parameters: size of the window t and the stride of the windows. The size of the window t is generally 2 or 3 and the stride s is taken equal to t, however some researchers proposed that taking s less than t reduces the error rate of the CNN by 0.4%.

The Max Pooling layer is introduced to increase the computation efficiency of the CNN by reducing the size of the data by a ratio of t, and by pooling maximum information from all the data in the pooling window, this triggers an association to particular feature orientations that could be useful in the classification stages.

With the combination of convolution and Max Pooling the first convolutions layer learns simple edge shapes with different orientation, the second convolutional layer learns to identify more complicated geometries such as intersections, angles. More meaningful semantics are leaned by the other convolution layers deeper in the network.

The Fully Connected layer, is generally the last layer before the classifier layer. It can be treated as a convolutional layer, however the learning is not restricted only to data in a small window but uses all the data in the previous layer.

The last fully connected layer is the classifier which calculates the class probabilities, the higher the probability the most probable is feature belongs to the class.

A.5.3 Optimisation techniques

Optimisation deals with the problem of finding a set of parameters that minimise the loss function. The most popular method to solve the loss function is the Stochastic Gradient

Decent, and the ADAM (ADAptive Momentum estimation) method used for large data sets with high dimensionality.

SGD is an incremental gradient descent and approximates stochastically the well-known gradient descent optimization method that minimises the loss function using the whole data set. SGD uses randomly select small batches of the training data set to achieve convergence. If L(θ) is the loss function that we are trying to minimise with respect to θ, the SDG updates the parameter θ as:

$$\theta \leftarrow \theta - \lambda \Delta \mathrm{L}(\theta) \qquad \textbf{(A 17)}$$

Where λ is the learning rate, and ΔL(θ) is the gradient of the loss function at the point of evaluation.

The ADAM method (see Kingma et all, 2014) is well suited for noisy large data sets, uses the first and second moment estimates of the gradients to compute the learning rates of each parameter, the ADAM updates of the parameter θ is given as:

$$\theta \leftarrow \theta - \frac{\lambda \hat{m}}{\sqrt{\hat{v}} + \varepsilon} \qquad \textbf{(A 18)}$$

Where λ is the learning rate, $\hat{m}$ is first moment estimate, and $\hat{v}$ is the second moment estimate, ε is a small number to avoid zero denominators.

TRL
Crowthorne House, Nine Mile Ride,
Wokingham, Berkshire, RG40 3GA,
United Kingdom
T: +44 (0) 1344 773131
F: +44 (0) 1344 770356
E: enquiries@trl.co.uk
W: www.trl.co.uk

ISSN

ISBN