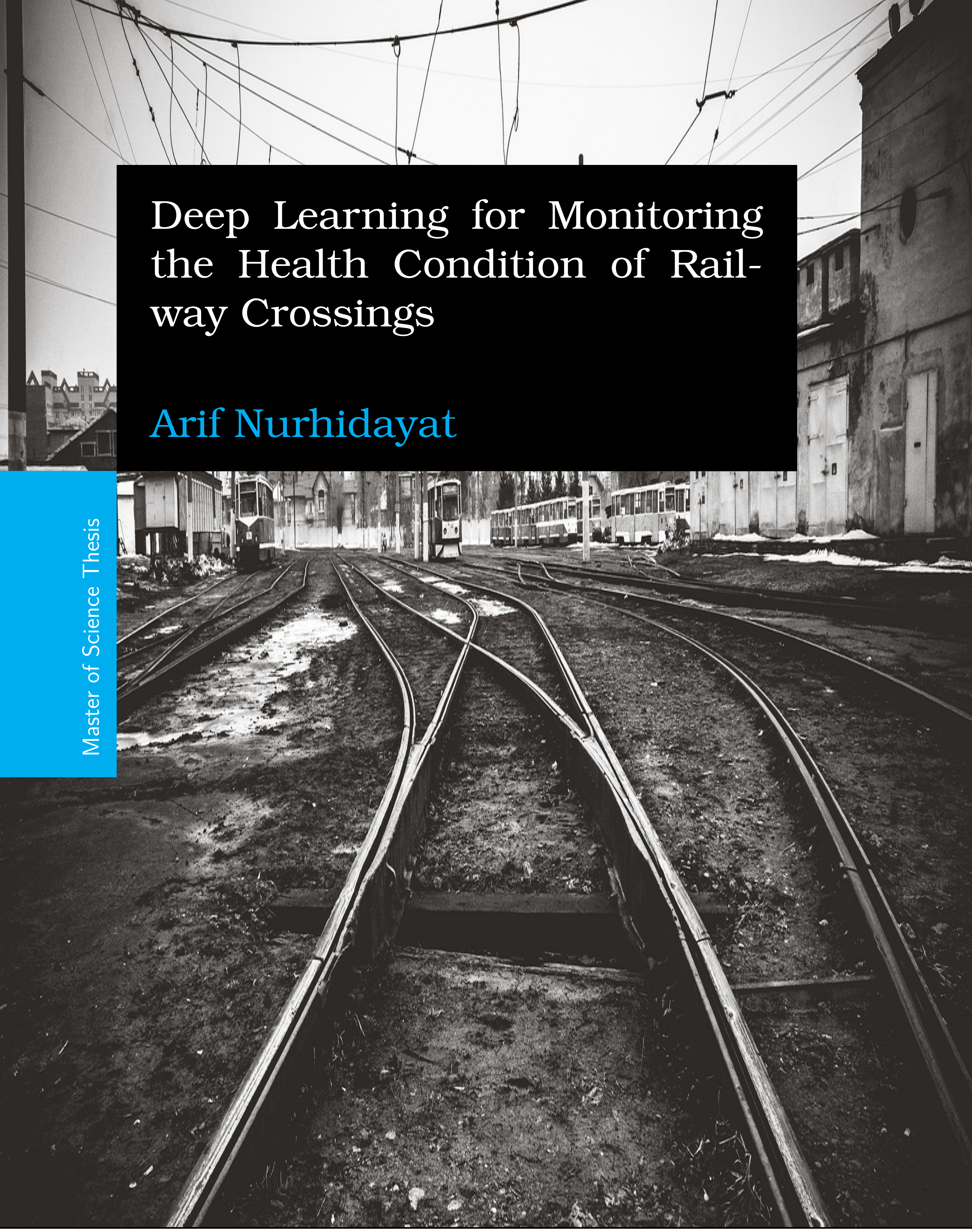
Deep Learning for Monitoring the Health Condition of Railway Crossings

Arif Nurhidayat

Master of Science Thesis



TU Delft Delft University of Technology

Delft Center for Systems and Control

Deep Learning for Monitoring the Health Condition of Railway Crossings

Master of Science Thesis

For the degree of Master of Science in Embedded Systems at Delft University of Technology

Arif Nurhidayat

January 19, 2018

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS) · Delft University of Technology

Abstract

Defects in railway crossings harm the quality of service provided by the railway operator. They can cause disruption time if they are detected in a late stage of deterioration, when urgent replacement or repair is the only solution.

In order to prevent defects from occurring, inspections are performed periodically. Inspecting railway crossings in areas with difficult access can be expensive and dangerous. Moreover, maintenance activities scheduled without actual knowledge of the condition of crossings can lead to unnecessary efforts, costs, and bad performance.

Nowadays, the use of more advanced instrumentation for monitoring the health condition of crossings is economically possible. However, many technical challenges are still open when converting data into knowledge for a better maintenance scheduling. Maintenance activities at the crossings can be more effective when the actual information and the forecast of their health conditions are employed, giving enough time for contractors to perform corrective actions. In this thesis project, a methodology to estimate and forecast the degradation level of a railway crossing based on vibration data recorded by several accelerometers installed on the crossing is proposed.

Due to various types of trains and other exogenous factors, responses obtained from accelerometers vary, making the problem of estimating the health condition of crossings is difficult to do analytically. In the methodology proposed in this thesis, deep learning algorithms are utilized to extract the condition of the crossing, and to provide the right features for the forecast of when defects on the crossing are about to be visually noticeable. The methodology is then evaluated using a dataset from the case study of a crossing located in Amsterdam. Using wavelet coefficients and the type of trains as features, the methodology can forecast defects on the crossing two months before the defects are spotted during the visual inspection.

Table of Contents

Acknowledgements

I would like to thank Indonesian Endowment Fund for Education for supporting me during my two-year study in TU Delft.

I also would like to thank my supervisors prof.dr.ir. B. De Schutter and Dr. A. Núñez Vicencio for their assistance during this thesis project, and ir. A. Boogaard for supporting me with suggestions and his experiment data.

Delft, University of Technology
January 19, 2018

Arif Nurhidayat

Chapter 1

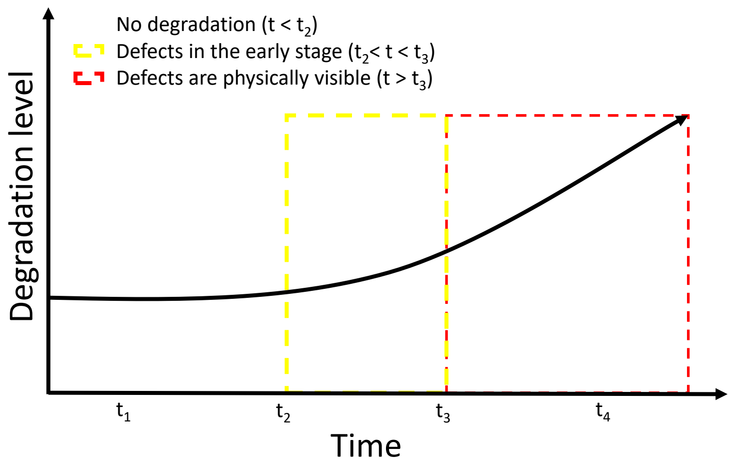
Introduction

Crossings are essential components in railway systems as they allow the trains to move from one line to another. Crossings are even more crucial in case of service disruptions as they can provide access to alternative routes. However, crossings are subject to high wheel-rail impact forces and vibrations during the passage of vehicles, which accelerate their degradation. It was reported in [1] that the railway infra managers usually spend 20% to 40% of their maintenance budget for repairing Switches and Crossings (S&C).

In order to ensure a safe operation of the crossing, contractors schedule timely visual inspections and measurements using various tools like handheld ultrasonics or specialized inspection vehicles when required. In case of detecting severe failures, corrective actions must be performed immediately.

Defects on the crossings often evolve fast, making preventive maintenance activities problematic if they are not scheduled wisely. Excessive inspections and maintenance activities can cause unnecessary spending on spare parts, time, and efforts that lead to inefficient expenses of the maintenance budget. However, not performing maintenance activities on time also causes a significant increase in the number of service disruptions. High costs of components and limited availability of specialized personnel make the maintenance problem even more complicated. If much time in advance is known, when crossings are about to fail, preventive actions can be scheduled more efficiently.

In this thesis, the detection of early-stage defects on crossings is investigated. Based on track-side vibration data and deep convolutional neural networks, a methodology that allows the monitoring of the health condition of crossing is proposed. As a case study, the dataset from measurements on a crossing located in Amsterdam is utilized. With the proposed methodology it is possible to detect the initiation of the failure of the crossing two months before the cracks are visible, giving the contractors additional time for planning their maintenance activities. Finally, the methodology can be extended for



age

Figure 1-1: The black line shows a typical trend of the degradation level. The black line inside the yellow box indicates the degradation level when defects are in the early stage, i.e. invisible to human eyes. The black line inside the red box indicates the degradation level when defects are visible to human eyes.

the health condition monitoring of other systems that can rely on vibration signals such as bearings and railway catenary systems.

1-1 Challenges of monitoring railway crossings

Vibration-based health condition monitoring is an interesting topic, particularly when this technology is applied on the railway crossing. Not many studies on this topic were found in the literature study. This lack of literature is an opportunity for performing research on this type of health condition monitoring system. Some of the reasons why this topic is challenging and has not been fully addressed in the literature are the following:

1. **Dynamics of railway crossings change slowly over time:** Defects on the crossing change the way it vibrates when a train passes over it. With no defects, the crossing vibrates at its natural frequencies. These frequencies change noticeably if the crossing is severely damaged and defects are usually visible. However, at an early stage when defects start to develop, and they are invisible to the human eyes, the changing frequencies are unnoticeable. As degradation of crossings is slow in the early stage as illustrated in Figure 1-1, detecting defects in this stage is more challenging than detecting defects once they are visible.

2. **Dominating wheel-crossing impact:** The vibration signal is not only affected by changing dynamics of the crossing but it is also very sensitive to wheel-crossing impact. Vehicle speed, wheel condition, among other factors, affect the measured

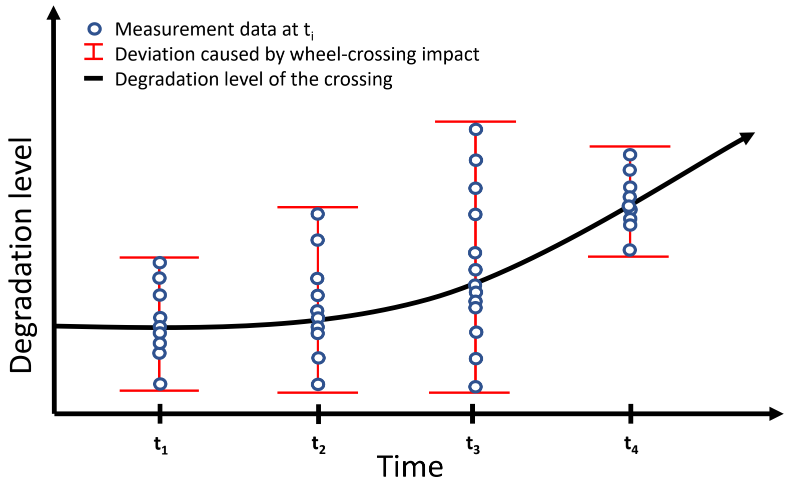


Figure 1-2: High deviation in the trend of the degradation level caused by the effect of the uncertainty of input signals.

signal. If changes of the signal due to wheel-crossing impact are more significant than changes of the signal caused by defects as illustrated in Figure 1-2, the detection task becomes more complicated.

3. **The size of data to be processed:** Detecting defects on the crossing in their early stage requires vibration data recorded at a high sampling frequency to capture short time transient responses. This kind of data needs a significantly more memory to save the same event compared to vibration data sampled at a lower rate. For example, 20 s of vibration data recorded by a 100 kHz accelerometer needs 8 MB of data (400 KB/s) compared to the one sampled by a 1 kHz accelerometer, which needs 80 KB of data (4 KB/s). This amount of data grows significantly with the number of sensors, measurements and crossings that need to be monitored. Since inspecting the evolution of the responses that slowly change over time requires a considerable amount of vibration data to be processed at once, this is a challenge we need to overcome in this thesis.

1-2 Research objectives

The primary objective of this research is determined as follows:

To develop a methodology that can estimate the health condition of a railway crossing based on its vibration data.

Moreover, considering possible challenges discussed in Section 1-1, the primary objective was divided into several sub-objectives as follows:

1. To develop a methodology that converts vibration signals into useful information. The use of state of the art methods is considered to accomplish this objective.
2. To compare the methodology with other state-of-the-art methods regarding performance, complexity, and robustness.
3. To find out the configuration of the chosen state-of-the-art method that can achieve the best performance.

The thesis is divided as follows. Chapter 1 introduces the primary problem, research challenges, objectives, and research scope. Chapter 2 discusses the literature study regarding railway crossings, conditions monitoring, and deep neural networks. Chapter 3 presents the proposed methodology for monitoring health conditions of railway crossings. Chapter 4 discusses implementation of the proposed methodology on the dataset from the case study of a crossing in Amsterdam. Chapter 5 presents conclusions, discussions, and further research. Figure 1-3 summarizes the outline of this report.

1-3 Research scope

The thesis focuses on developing a methodology for condition monitoring as discussed in Section 1-2. Since different approaches can be used to achieve the objectives, the scope of this research should be limited.

1. **Limiting exogenous factors:** Dynamical responses of the railway crossings are affected by many factors. In this thesis, factors that affect the responses are limited to the wheel-crossing impact forces of passing trains and dynamical changes of the crossing due to defects. Other factors such as the effect of sleepers, ballasts, and others are not considered because of the information about them is not available.
2. **State of the art approaches:** While many state-of-the-art methods might also be applied to solve the problem in this research, the focus is on Convolutional Neural Networks (CNN) algorithms. The reason is that this research requires the method to perform classification and regression tasks from high dimensional input data, which makes the CNN the most suitable approach [2]. Moreover, many CNN algorithms have succeeded to perform similar tasks with very good performance [3, 4, 5, 6].
3. **Number of crossings:** For generalization purposes, data measured from multiple and different crossings might be desirable. However, due to strict regulations on operational crossings, it was not possible to obtain data from other crossings. The proposed methodology is generic, applicable to new cases of crossings, but the results of the case study are limited to the condition of that specific crossing.

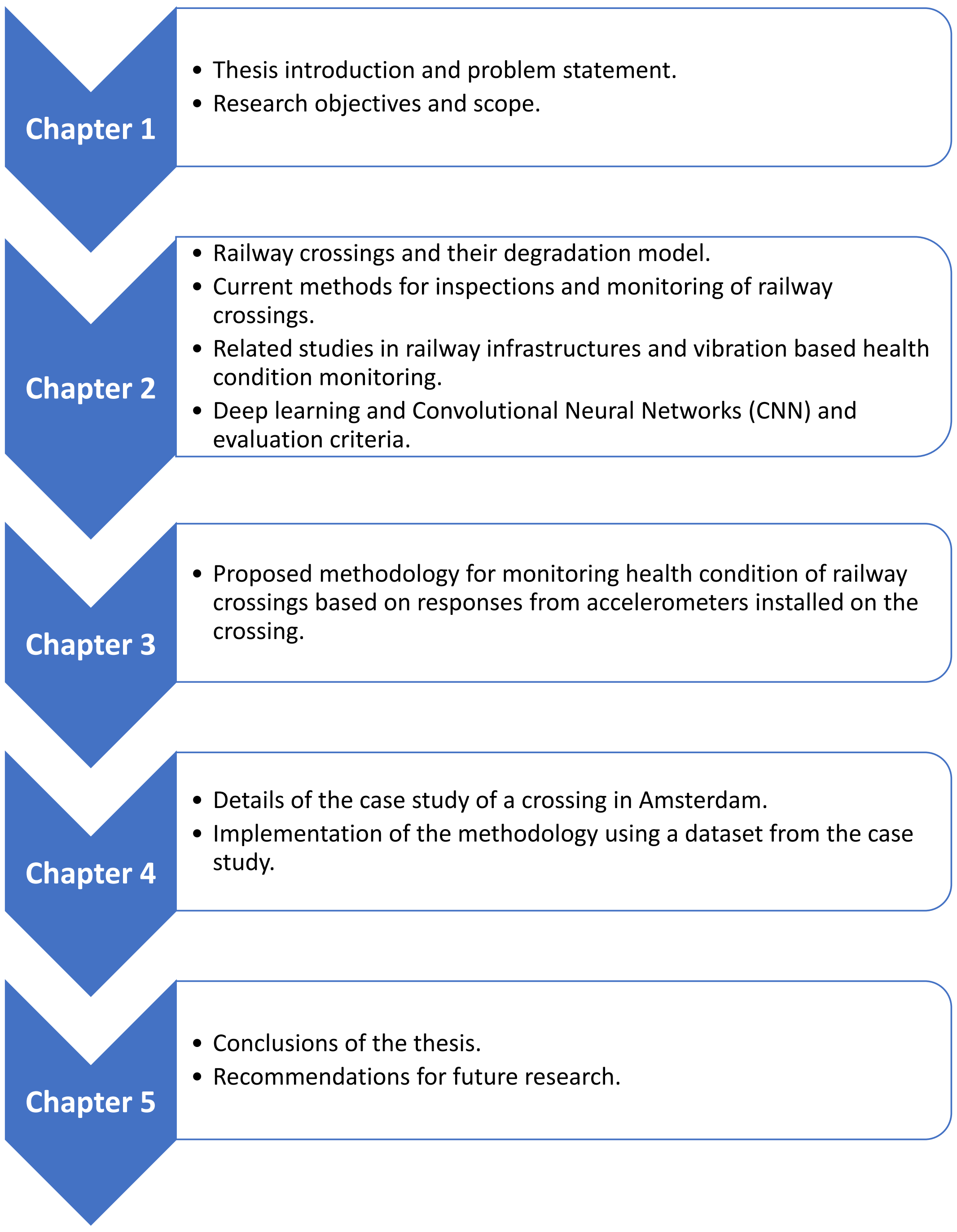

Chapter 1
Thesis introduction and problem statement.
Research objectives and scope.
Chapter 2
Railway crossings and their degradation model.
Current methods for inspections and monitoring of railway crossings.
Related studies in railway infrastructures and vibration based health condition monitoring.
Deep learning and Convolutional Neural Networks (CNN) and evaluation criteria.
Chapter 3
Proposed methodology for monitoring health condition of railway crossings based on responses from accelerometers installed on the crossing.
Chapter 4
Details of the case study of a crossing in Amsterdam.
Implementation of the methodology using a dataset from the case study.
Chapter 5
Conclusions of the thesis.
Recommendations for future research.

Figure 1-3: Outline of the thesis report.

Chapter 2

Literature Study

In this chapter, the topics discussed are: railway crossings and their degradation, condition monitoring of railway infrastructures, and deep learning algorithms.

2-1 Introduction to railway crossings

A railway crossing is part of a turnout that ensures the continuity of two intersecting routes, as illustrated in Figure 2-1. The crossing consists of four components: check rails, wing rails, flangeway opening, and crossing nose. Check and wing rails ensure the safe passage of the axle by guiding the wheel. The flangeway opening located in front of the crossing nose provides a path for the wheels of trains to depart safely toward one of the directions. The flangeway opening, because it causes discontinuity of the rail, causes wheels of trains that pass the gap to jump and hit the surface of the crossing nose. For this reason, the railway crossing is also called the "frog". As a consequence, the surface of the crossing nose suffers from high impact load that causes damages [7].

Various kinds of damages can be found in crossings like cracks, spalling, plastic deformation, and shelling [9]. These damages can develop excessively because of the continuous usage of the crossing. Nowadays, most defects originate from the surface, in the contact area between wheel and crossing. Early stage defects are not visible to human eyes because they grow into the subsurface, as shown in Figure 2-2. Moreover, the deformation of the crossing in the early stage defects is minimal (on the scale of µm). If the maintenance engineer is unaware of their existence, cracks will develop and they become a threat to the safety of the operation. As a consequence, as soon as defects are detected, the faulty crossing has to be replaced. In the worst case scenario, undetected defects in crossings can cause failures, which could lead to trains derailment. In order to

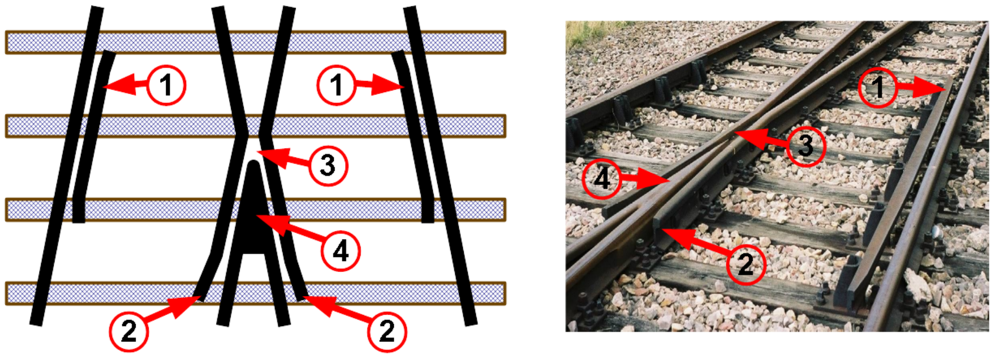


Figure 2-1: Illustration of railway crossing: 1. check rails; 2. wing rails; 3. flangeway opening; 4. crossing nose [8].

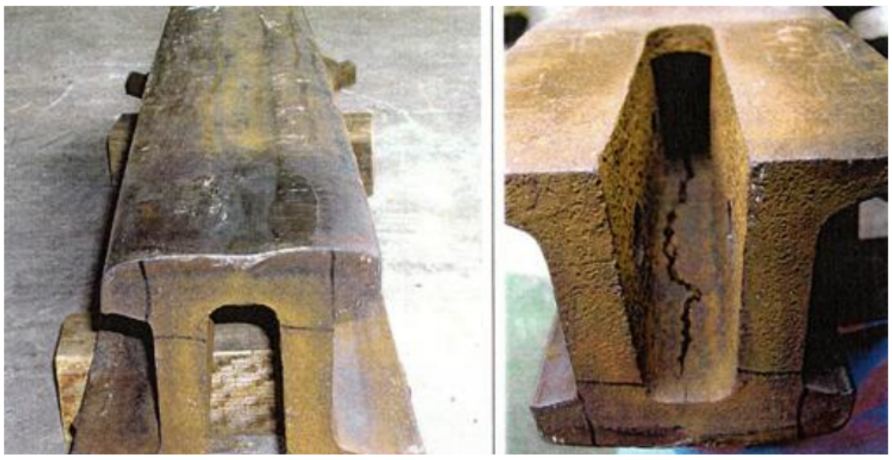


Figure 2-2: Subsurface crack on the crossing [9].

prevent this to happen, several studies have been performed to understand and model the behavior of degradation in railway crossings.

2-2 Degradation model in railway crossings

Modeling the degradation of railway crossings is done for several reasons: to understand the factors contributing to the degradation, and to predict when railway crossings will fail in order to perform more efficient maintenance. Two approaches are usually used for modeling the degradation of railway crossings: regression methods applied to historical data (maintenance database), and dynamical models of crossings via simulation of the wheel-crossing contact.

In the approach using a maintenance database, the relationship between the level of degradation and different parameters is found using a regression function. In [1], it was shown that the train load and the complexity of the crossing play significant roles in the failures of crossings. In another similar study [10], a function to approximate the degradation level of the crossing from several parameters was determined using linear regression. It was found that the complexity of crossings and the operation time give most statistical effects on the degradation level. Additionally, a study in [11] concluded that we can use a three-parameters Weibull distribution to find the relationship between failures in the crossings and the operation time.

In the other approach using dynamical models [12], it was concluded that increasing axle load leads to increasing degradation. In another study [13], a numerical procedure for analysis of rolling contact fatigue crack initiation and fatigue life prediction for railway crossing was proposed. A 3D finite-element model was created based on the drawings from the manufacturer. The model was developed to simulate the dynamic response of wheels passing the crossing. The simulation showed that the stress is high in the surface area of crossing due to the impact force. Additionally, this study also concluded that most of the damages occur in the area where the stress is high, and that the initiation of fatigue cracks can be predicted based on the passing axles. In [14], a 3D finite-element model of wheel-rail impact at railway crossings was developed. The study showed that the actual dynamics of the crossing is more complicated than usually assumed in the literature. The model suggests that damage analysis and structure optimization could lead to the design of new and better crossings.

The degradation level of the crossing can be estimated with both approaches, allowing the maintenance engineers to set predefined priorities in scheduling the maintenance activities. However, since the crossing can fail anytime during the expected lifetime, it is essential to incorporate the actual information from the field to the models so that the expected lifetime can be predicted with better accuracy (i.e., closer to the real condition of the crossing).

2-3 Current methods for checking the health condition of railway crossings

Currently, a lot of railway operators still prefer periodic inspection over condition monitoring. The inspections are performed either by inspectors or inspection vehicles, as explained in [15]. The inspectors visit the crossings on a monthly or weekly basis to conduct a visual and measured check. For regular checks using inspection vehicles, various sensors such as high-definition cameras, lasers, ultrasonics, and eddy currents are utilized together with GPS to tag the location of the inspected components. The inspection vehicles are utilized because they are safer (no personnel on track) and faster (they can inspect up to 80 to 100 km of track distance per hour) than inspectors. However, crossings are complicated structures requiring specialized devices to measure their condition with a more significant accuracy. Thus, a site visit performed by personnel is still preferred for a detailed measurement such as a hammer test to obtain dynamic properties of the structures, ultrasonics or phase array tests to look for hidden cracks, and 3D scan to get an accurate profile of the crossing.

2-3-1 Hammer excitation test

A hammer excitation test can be performed to obtain dynamic properties of the structures for measuring performance [16]. In railway systems, it is used for rails and train wheels. The hammer test is carried out by hitting the track with a special hammer equipped with a force sensor to apply a vertical impulse load that produces track vibration. The resulting responses are recorded using accelerometers attached to various locations on the track. The parameter of the track can be obtained by converting the recorded signal by means of Fast Fourier Transform (FFT) into the Frequency Response Function (FRF) while assuming linear behavior of the track. The FRF $H(s)$ relates the fourier transform of the input force $U(s)$ and the fourier transform of the ouput response $Y(s)$ as

$$H(s) = \frac{Y(s)}{U(s)} \tag{2-1}$$

With defects in the track, dynamics of the crossing are changed. Therefore, the failure mode can be noticed from $H(s)$ which is not sensitive to the changes of $U(s)$.

This principle can be adopted for monitoring railway crossings by installing accelerometers to capture the transient responses induced by train wheel-crossing impacts [17, 18]. In this case, the wheels act as hammers.

2-3-2 On board monitoring of railway infrastructures

Several approaches such as vehicle-based [19, 20], axle-box acceleration based [21, 22, 23, 24], and online monitoring [25, 26] have been proposed for condition monitoring of railway infrastructures. In online monitoring, various kinds of sensors are installed on stationary places near the infrastructures so that the monitoring can be done in real time. In vehicle-based monitoring, the sensors are put either in dedicated inspection vehicles or in-service trains. Compared to the online monitoring, using inspection vehicle or in-service trains may be cheaper as they can be used for different infrastructures without additional cost. However, they may reduce the capacity of the monitored components, and they are not capable of providing real-time information as the sensors keep moving in different locations.

Different sensors have been proposed in recent studies for railway infrastructures condition monitoring that may be utilized for railway crossings such as ultrasonics [27], lasers [28], high-definition cameras [25, 29], eddy current sensors [30], and accelerometers [23, 31]. Accelerometers have several advantages over the other sensors. They are tougher and more robust compared to lasers or high-definition cameras. The lens of high-definition camera and lasers have to be cleaned periodically to maintain their measurement accuracy. Moreover, accelerometers are also not affected by other objects near them. In contrast, other sensors such as high-definition cameras, lasers, and ultrasonics do not work when they are blocked by other objects. Lastly, accelerometers are cheaper than the other sensors.

Accelerometers produce vibration data that needs signal processing involving frequency domain analysis, filtering, and other mathematical operations before it can be utilized to present meaningful information to the users. A drawback might appear when the signal to noise ratio of the vibration data is too high, which makes the processing even more complicated to extract the right features.

2-4 Vibration-based condition monitoring

Using vibration signals in infrastructure monitoring has become a common practice. Vibration analysis is widely accepted as a tool to monitor operating conditions as it is considered as a reliable and nondestructive testing method. It also permits continuous monitoring. In early usage, engineers used the amplitude of vibration to set up various stages of maintenance [32]. This method is intuitive, and the data gathered in field surveys show a good agreement with the trend illustrated in Figure 2-3. For example, the study in [33] showed that switches that generate fewer noise and vibrations (due to lubrication) have 50% more lifetime. Once faults are identified, the operator should closely monitor specific features so that the rate of deterioration caused by the faults can be revealed. In the case of railway crossings, this method is too general since vibrations on crossings are induced by different kinds of inputs that make characteristics of fault trends differ substantially.

Explicit and implicit methods are available for vibration-based condition monitoring. In explicit methods, faults are associated with the dynamics of the system. According to [34], changes in train loads, sleepers, ballast, and the type of soil can cause signals to vary. A mass-spring-damper model can be constructed to isolate and analyze the faults. Then, the vibration signals from the model can assist to identify the features indicating different types of faults. In implicit methods, features indicating the faults are directly inspected from the signal, e.g. the studies in [35, 36]. The implicit approach is considered to be more practical since creating a dynamical model is not always possible.

2-4-1 Features used in implicit methods

Since a vibration is a time-varying signal, it is difficult to find features indicating the faults without transforming the signal into something else that is easier to analyze. Several available methods such as statistical operations, time to frequency domain transformation, signal source separation, feature compression, and other signal processing

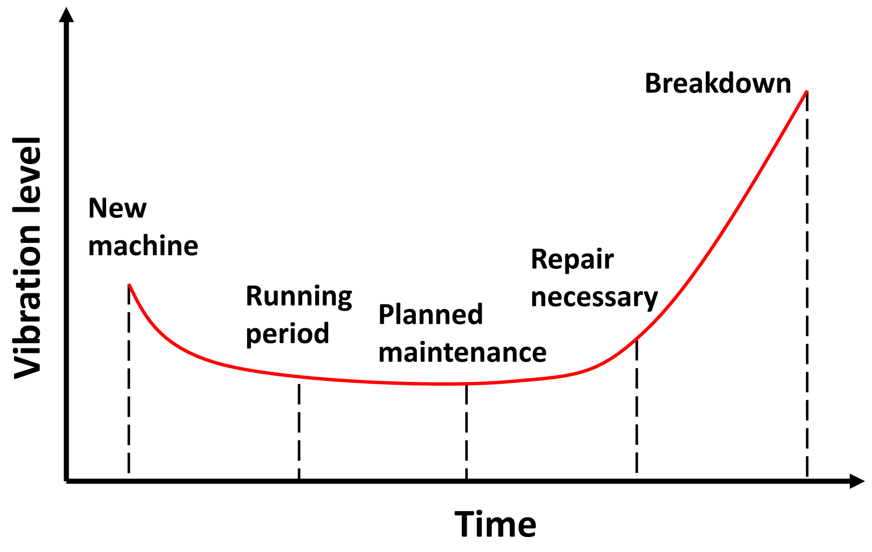


Figure 2-3: Typical trend of temporal vibration signal showing the rate of deterioration [32].

algorithms can be used to change the time domain signal into simpler representing values. Among others, two popular methods that are widely used are the wavelet transform and statistical operations.

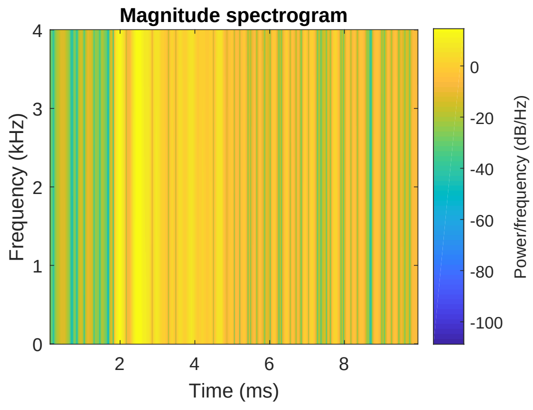
Wavelet transform

Similar to the short-time FFT, the Continuous Wavelet Transform (CWT) also converts a signal from the time domain to its representation in the frequency domain [37, 38, 39]. In the case of the short-time FFT, a sliding window function $g(t)$ is utilized to sample a nonperiodic signal $s(t)$ around the time τ [40]. The short-time FFT $S_{\text{fft}}(\tau, \omega)$ is expressed as

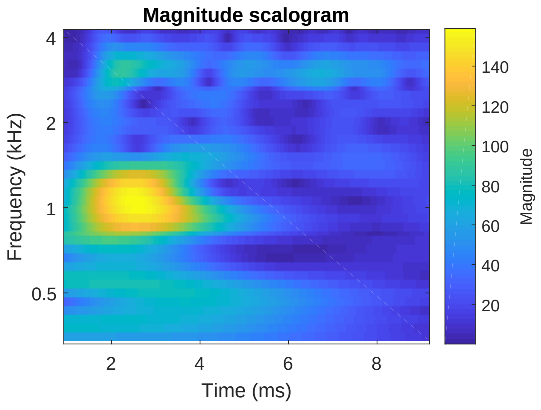
$$S_{\text{fft}}(\omega, \tau) = \int s(t) g(t - \tau) e^{-j\omega t} dt \tag{2-2}$$

The short-time FFT produces a uniform distribution across all frequency bands. Consequently, inappropriate selection of τ will cause the results in lower frequencies bands too narrow or higher frequencies bands too coarse. The CWT solves this problem as it is specially designed to capture transient dynamics. The CWT $C(b, a)$ can be expressed as

$$C(b, a) = \frac{1}{\sqrt{|a|}} \int \psi^{\star} \left(\frac{t - b}{a} \right) s(t) dt \tag{2-3}$$



(a)



(b)

Figure 2-4: Results of time-frequency transformations from a $10ms$ transient response: (a) short-time FFT and (b) CWT.

where $\psi(\cdot)$ is the wavelet function, and both a and b are scaling variables. Using the Morlet's wavelet function with a particular radial frequency ω_o

$$\psi(t) = \pi^{-\frac{1}{4}} \exp(i\omega_o t) \exp\left(-\frac{t^2}{2}\right) \tag{2-4}$$

the CWT as a function of time and frequency domain can be obtained as the following:

$$C(f,t) = C(b,a)\{s(t)\}; \quad f = \frac{a}{\omega_o 2\pi}; \quad t = b \tag{2-5}$$

The results of short-time FFT and CWT applied on the wheel-crossing impact forces signals are shown in Figure 2-4 (a) and (b) consecutively. Both figures clearly show that the CWT produces a better resolution, especially in the higher frequency domain. Wavelet coefficients can be used as features directly, or they can also be processed again by performing statistical operations on their particular time or frequency domain to yield a better representation of signals, as shown in [41, 42, 43].

Statistical values from time domain signals

Statistical operations can be performed directly on the time domain signal to reduce its dimension. The premise behind this method is that a different shape of signals has different statistical values. Therefore, the statistical values can be used to represent the signal. The advantage of this method is that it is easy to implement and may work well if the features indicating the faults are observable from the signal.

Statistical features were used in several studies. In [31], the maximum amplitude and the wavelength of the first peak were utilized to determine severity of defects in insulated rail joints. In other study [44], statistical values such as variance, maximum amplitude, number of peaks, average amplitude, area under the peaks, etc were utilized to classify the types of trains.

In the case of crossings, where features indicating the faults appear during short periods of the time or lie in a higher frequency band with a small magnitude, statistical features will not work well. This is because of the indicating features are too small compared to other peaks that dominate the signal. In this case, statistical operations such as average, maximum, minimum, median, mode, standard deviation, and variance might not coincide with features that indicate the faults.

On the other hand, the CWT is suited for features which appear on the particular frequency even with the small magnitude because the CWT expand the signal into its representation in both the time domain and the frequency domain. In this case, the features are more noticeable as they are located in the particular position of the time-frequency plane.

2-4-2 Recently proposed method for monitoring health conditions of railway crossings using vibration data

In [23], Axle Box Acceleration (ABA) measurements were used to evaluate a crossing with two types of degradations: uneven deformation between the wing rail and the crossing nose, and local irregularity in the longitudinal slope of the nose. The wavelet transform was applied on the acceleration signals to derive features. Several operations applied on the wavelet coefficients like the power of two to produce the wavelet power spectrum, and a weighted sum in the frequency domain to produce the scaled-averaged wavelet power for improving quantification in the certain frequency range.

The results show that at the frequency of 600 Hz, the values of wavelet power spectrum were high at the distance of the wheel-crossing contact (uneven deformation) and 50 mm from the crossing nose (local irregularity) after the crossing was degraded. While the research utilized ABA measurements, it is a challenge of this thesis to detect crossing defects with trackside acceleration measurements.

With vibration-based condition monitoring, we can monitor different kinds of railway infrastructures like the railway track, train wheels, Switches and Crossings (S&C), etc. Moreover, with the current development of hardware and learning algorithms, e.g., the deep learning, the condition monitoring can be automated. We can design the condition monitoring system that is not only capable of giving the actual health condition of the infrastructure, but also predicting when it is going to fail. In this scenario, a particular type of deep learning architecture called the Convolutional Neural Networks (CNN) is used to locate the features indicating the defects and find a nonlinear function associating the features with the degradation model.

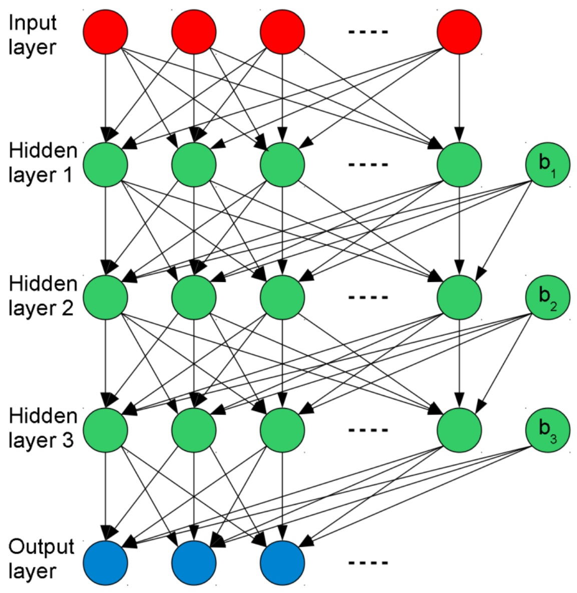


Figure 2-5: A multi-layer perceptron with three hidden layers.

2-5 Introduction to the deep learning algorithms

Deep learning can be described as a multi-layer perceptron with many (usually more than two) hidden layers [45] as illustrated in Figure 2-5. A multi-layer perceptron consists of three different layers represented by a column vector belonging to input, hidden, and output layers. In each hidden layer, a column vector X^{l-1} of the input (or the output of previous hidden layer) is multiplied by weight matrix W^l and added by a bias vector B^l. The product is then fed into an activation function $f(\cdot)$ as described by

$$Z^l(X^{l-1}) = W^l X^{l-1} + B^l \tag{2-6}$$

$$X^l = f(Z^l(X^{l-1})) = \max(0, Z^l(X^{l-1})) \tag{2-7}$$

where the l indicates the position of the layer (i.e. the l is the current layer, and the $l-1$ is the previous layer). In the output layer, similar vector operations are performed with a different activation function $f(\cdot)$ as

$$f(X) = \frac{\exp\left(Z_j^l\right)}{\sum_{j=1}^{l} \exp\left(Z_j^l\right)} \tag{2-8}$$

where j denotes each element of the column vector $Z^l(X^{l-1})$. For a classification task, the class of vector input is decided based on the highest score generated by the activation

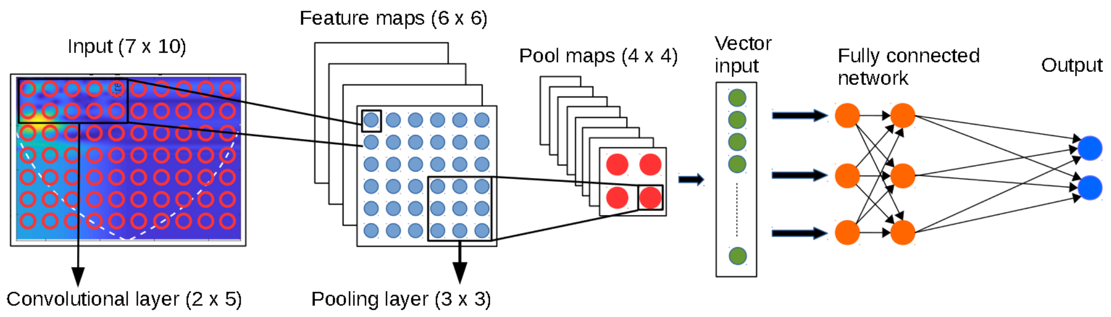


Figure 2-6: Convolutional Neural Networks (CNN)

function in Equation 2-8. For a regression task, the score of each element of the vector output Z_j^l is used to determine how close the vector input is to each class in the vector output.

With high-dimensional input data, a fully connected multi-layer perceptron is not suited for performing both the classification and the regression tasks as the size of the weight matrix W is quadratic to the dimension of the input data X. For this kind of task, CNN algorithms are recommended.

2-5-1 CNN architecture

CNN is a deep learning architecture that is suitable for recognizing patterns from high-dimensional input data such as the wavelet coefficient matrix as shown in Figure 2-4 (b). The CNN algorithm works by deriving features from the input during the training, and predicts the output class based on the contribution of each feature. Different CNN algorithms have been deployed for image tagging [2], sound classification [5], and vibration-based fault predictions [3, 4, 6, 46]. As illustrated in Figure 2-6, a CNN architecture consists of convolutional layers, general pooling layers, and a fully connected multilayer perceptron. The convolutional and the pooling layers produce features, which are saved in matrices called feature maps and the pooling maps. Both feature and pooling maps act as nonlinear filters magnifying and reducing the effect of specific areas of the input related to the output. The fully connected multilayer perceptron acts as either the classifier or the regressor, which processes the vectorized output of the last pooling layer (not the prediction).

A convolutional layer is a shared weight multi-layer perceptron that produces a feature map from the input vector with a sliding window mechanism so that the size of the weight matrix in the convolutional layer is significantly reduced compared to the one using a fully connected multi-layer perceptron. For example, assuming that the input contains 7×10 elements, using a sliding windows mechanism, a convolutional layer consisting of 2×5 of weights is enough to produce a 6×6 feature map as shown in Figure 2-6. In contrast, a fully connected multi-layer perceptron needs at least 2520

$(7 \times 10 \times 6 \times 6)$ weights (all nodes must be connected) to produce a 6×6 feature map from an input consisting of 7×10 elements (all elements of the input and feature map must be connected). In order to produce N feature maps, N convolutional layers are needed. As feature maps act as learning filters, having more convolutional layers mean extracting more features from the vector input.

A pooling layer is usually used to reduce the size of feature maps by obtaining representing values based on operations such as max, average, or other operations. In Figure 2-6, a 6×6 feature map is reduced into a 2×2 pooling map using a 3×3 pooling layer. In this case, 9 elements of a feature map are represented by 1 element of a pooling map using max operation. After the pooling operation, the matrix can be flattened as a vector input for the fully connected multi-layer perceptrons, or it can be fed into convolutional layers again to increase the depth of the CNN architecture.

The key performance of CNN lies on the configuration of its convolutional and pooling layers, since both of them determine the number of features. However, in a real-life application, the complexity of the CNN architecture must be restricted due to the hardware constraints. This trade-off must be settled so that the performance of the CNN is not below the desired standard while maintaining the algorithm's level of complexity.

2-5-2 CNN for monitoring the railway infrastructures

CNN algorithms have been used for monitoring the health condition of railway infrastructures [47, 29]. In [47], eight separated convolutional networks were used to extract nonlinear features from the wavelet coefficients produced by the signals from eight strain gauges. The strain gauges were installed between the sleepers. Then, the final class probabilities (defects and normal) were obtained by connecting all the convolutional layers into two fully connected layers. The experiment achieved 84% of accuracy.

In another study, a concept of binary features was introduced in [29]. The idea of this study was to perform multiple classification tasks in a single image by recognizing the existing objects in the image using CNN for further detection of material damages. Full deep convolutional networks were utilized to produce binary features consisting of material and fastener classes. The architecture consisted of three shared convolutional layers where the output of the third layer was fed to two separated convolutional layers. One separated convolutional layer was used for the material classification, and another convolutional layer was used for the fastener classification. With this architecture, the predictions achieved 95% accuracy.

Other applications include detection of squats based on video images [48], and the detection of missing elements in the catenary system [49].

2-5-3 Performance measures

Several widely used performance measures are available for evaluating the performance of learning algorithms. For the classification task, four measures can be utilized such as overall accuracy J_{a}, recall J_{r_i}, precision J_{p_i}, and specificity J_{s_i} [50]. The overall accuracy is expressed as

$$J_{\mathrm{a}} = \frac{1}{C} \sum_{i=1}^{C} \frac{\mathrm{tp}_i + \mathrm{tn}_i}{\mathrm{tp}_i + \mathrm{tn}_i + \mathrm{fp}_i + \mathrm{fn}_i} \tag{2-9}$$

where C is the total number of classes, and i denotes the index of the class. Other measures such as recall, precision, and specificity are evaluated per class i as

$$J_{\mathrm{r}\_i} = \frac{\mathrm{tp}_i}{\mathrm{tp}_i + \mathrm{fn}_i} \tag{2-10}$$

$$J_{\mathrm{p}\_i} = \frac{\mathrm{tp}_i}{\mathrm{tp}_i + \mathrm{fp}_i} \tag{2-11}$$

$$J_{\mathrm{s}\_i} = \frac{\mathrm{tn}_i}{\mathrm{tn}_i + \mathrm{fp}_i} \tag{2-12}$$

where tp_i indicates the number of correct positive predictions (the member of the class i is predicted as the member of the class i), tn_i indicates the number of correct negative predictions (the member of the other class is not predicted as the member of the class i), fp_i indicates the number of incorrect positive predictions (the member of the other class is predicted as the member of the class i), and fn_i indicates the number of incorrect negative predictions (the member of the class i is detected as the member of the other class). These variables are summarized in Table 2-1.

Table 2-1: Confusion matrix: labels in the row direction show the actual data, and labels in the column direction show the predicted data

	Predicted: True	**Predicted: False**
Actual: True	tp_i	fn_i
Actual: False	fp_i	tn_i

For the regression task, Mean Squared Error (MSE)

$$\mathrm{MSE}_{y_i \in [0,1]} = \frac{1}{L} \sum_{i=1}^{L} (y_i - \hat{y}_i)^2 \tag{2-13}$$

is widely used to measure the overall closeness of the result $\hat{y}_i$ with the reference value $y_i \in [0, 1]$, with L the number of data. Otherwise, if $y_i \notin [0, 1]$,

$$MSE_{y_i \notin [0,1]} = \frac{1}{L} \sum_{i=1}^{L} \left(\frac{y_i - \hat{y}_i}{y_i} \right)^2 \tag{2-14}$$

is utilized. In order to evaluate the complexity of this methodology, the computation time and the learning time are measured. Additionally, the number of trainable parameters (the number of weights) is also considered.

2-6 Summary

This chapter introduced the background knowledge regarding this thesis project, including the monitoring of the health condition of railway crossings and the deep learning.

In the first part of the chapter, railway crossings are discussed. A railway crossing ensures the safety of the train to move from or to a particular direction of two intersecting routes. The railway crossing consists of four components: check rails, wing rails, flangeway opening, and crossing nose. The flangeway opening causes wheels of the passing train to jump and hit the crossing nose, which causes damages [7] that can develop excessively [9].

Degradation models have been constructed to find the expected lifetime of the crossing. These models are either constructed from the maintenance database [1, 10, 11], or the dynamical model of the wheel-crossing impact [12, 13, 14, 13]. Since the crossing can fail anytime during the expected lifetime, it is essential to incorporate the actual information from the field to the models so that the expected lifetime can be predicted with better accuracy (i.e., closer to the real condition of the crossing).

In order to obtain the real condition of the crossing, the railway operator performs periodic inspections and various measurements. These measurements include hammer tests to obtain dynamic properties of the structures, ultrasonics or phase array tests to look for hidden cracks, and 3D scan to get an accurate profile of the crossing.

In the second part of this chapter, the vibration-based condition monitoring is discussed. The vibration-based condition monitoring is a common practice to monitor operating conditions and faults of infrastructures, as it is considered as a reliable and nondestructive testing method. Two approaches in the vibration-based condition monitoring are the explicit and the implicit approaches. In the explicit approach, the operating condition of the infrastructure is associated with the dynamics of the system [34]. In the case of the implicit approach, features indicating the faults are directly inspected from the signal [35, 36]. The implicit approach is more practical since creating a dynamical model of the infrastructure being monitored is not always possible.

Among many features which are available for the implicit approach, two of them are widely used: Continuous Wavelet Transform, and statistical operations. The wavelet transform converts the vibration signal into its representation in the time-frequency domain, and the statistical operations extract the representing values by performing statistical operations directly from the signal. In the case of monitoring railway crossings, features obtained from the CWT is more suitable as values that indicate the defects are located in a particular position of the time-frequency plane.

In the third part of this chapter the Convolutional Neural Networks (CNN), which is suitable for recognizing patterns from high-dimensional input data such as the wavelet coefficient matrix, is introduced. A CNN architecture consists of convolutional layers, general pooling layers, and a fully connected multilayer perceptron. The convolutional and the pooling layers extract features from the input, and the fully connected multilayer perceptron perform either the classification task or the regression task.

Different CNN algorithms have been deployed for image tagging [2], sound classification [5], and vibration-based fault predictions [3, 4, 6, 46]. In the case of monitoring railway infrastructures, CNN algorithms have been used for classifying railway track materials for the further detection of defects [29], detecting the existence of squats on the track [47, 48], and the detection of missing elements in the catenary system [49].

Chapter 3

Methodology for Monitoring Railway Crossings Using Deep Learning

In this chapter, the focus is on the condition monitoring to predict defects in the crossing considering the early evolution of the degradation. The methodology estimates the degradation level of crossings based on the inputs from trackside acceleration signals. The methodology is divided as follows. Section 3-2, the processing for detecting and extracting transient responses. Section 3-3, relying on deep Convolutional Neural Networks (CNN) algorithms, features extraction is conducted. Section 3-4, the estimation of the degradation level of the crossing is described.

3-1 Problem description

The proposed methodology adopts a similar principle to the hammer excitation test. Several accelerometers are installed in the crossing, and the wheel-crossing impact forces are used as the excitation signal (instead of a hammer). In order to understand the theoretical difference with respect to the hammer excitation test, consider the Frequency Response Function (FRF)

$$H(s) = \frac{Y(s)}{U(s)} \tag{3-1}$$

which represents dynamical properties of the crossing. The $Y(s)$ is the Fourier transform of the acceleration signal due to the Fourier transform of the input force $U(s)$. In the case of the hammer test, $U(s)$ and $Y(s)$ are measured. In contrast, the Fourier transform of the input force $U(s)$ is not measured in the trackside measurement. As a consequence,

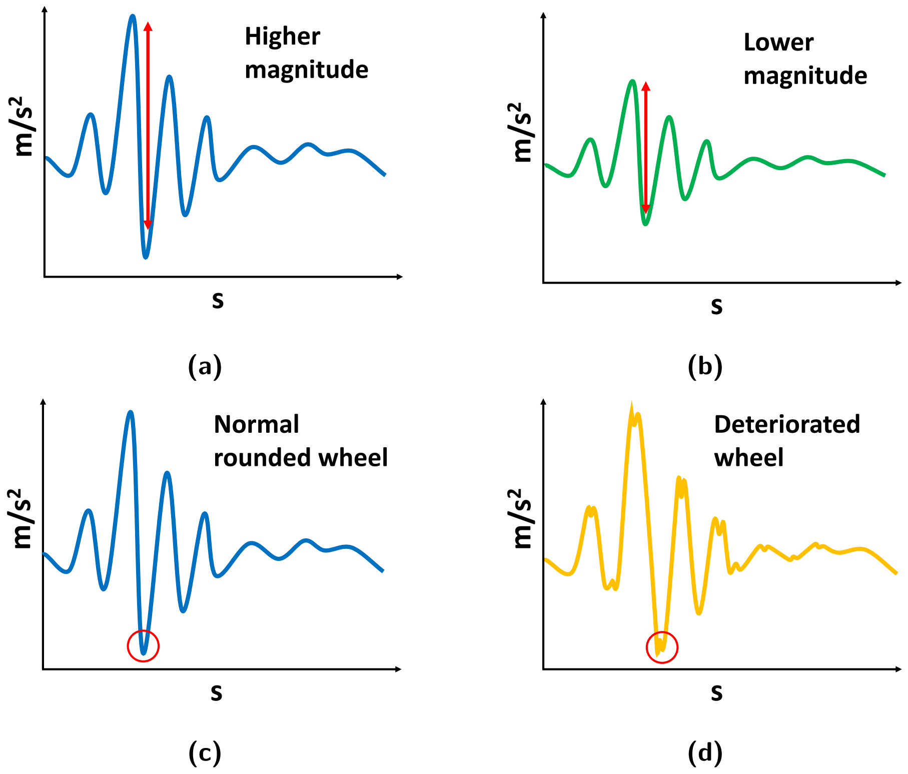


Figure 3-1: Effects of different trains passing the crossing on transient responses: (a) higher magnitude response induced by a heavy train; (b) lower magnitude response induced by a light train; (c) smooth response induced by a normal rounded wheel; (d) distorted (higher frequency) response induced by a deteriorated wheel.

the $H(s)$ cannot be obtained. The analysis is then performed only based on the output $Y(s)$, which is affected by both dynamical properties of the crossing $H(s)$, and the unknown input $U(s)$.

Wheel-crossing impact forces $U(s)$ are influenced by mass, speed, and moving direction of the train. A heavy train that moves very fast will apply a higher impact force compared to a lighter train that moves slower. Trains facing the crossing in different directions also apply different forces. A train that moves toward the crossing nose from the switch has a higher impact force compared to a train that moves in the opposite direction [51]. The influence of wheel impact on the signal is illustrated in Figure 3-1 (a) and (b). In the case of wheel condition, the deteriorated surface of the wheel can induce higher frequency responses similar to the case of the surface defects on the crossing [52]. The influence of the condition of the wheel on the signal is illustrated in Figure 3-1 (c) and (d). Since different factors cause $U(s)$ to vary, it is challenging to estimate the degradation level of the crossing only from $Y(s)$.

The arrangement of accelerometers in this proposed method is illustrated in Figure 3-2, where each one produces different forms of acceleration signals. The accelerometer placed nearest to the location of the highest wheel-crossing impact generates signals with the highest magnitude, and the one placed furthest to the crossing nose yields signals with the lowest magnitude. In the case of trains coming from the switch toward the crossing nose, peaks of the signals are first sensed by Accelerometer a. This condition is illustrated in Figure 3-3. In the opposite direction, the peaks are first sensed by Accelerometer c.

In order to study the early stage of degradation level, accelerometers are set to record the responses in high frequency sampling (100 kHz). As a result, the amount of vibration signals is very large. One hour of measurement would easily yield into 1.4 GB of data approximately. However, not the whole signal signal contains useful information. When the train wheel is not passing over the crossing, there is no source exciting accelerations. Thus, the recorded signals contain meaningless information. In order to keep the memory usage low without losing information, only parts of signals containing important information are kept. In the case of signals illustrated in Figure 3-3, only transient responses induced by wheels impact (highlighted by orange boxes) are kept. For this purpose, preprocessing is conducted to the whole measured signals.

Transient responses $Y(s)$ are affected by both varying inputs $U(s)$ and the evolution of defects that changes the dynamical properties of the crossing $H(s)$, as discussed before. Therefore, important features should be extracted from the transient responses. Two types of features are considered in this methodology. First, as the trend is affected by varying inputs, features such as the type of the train are utilized for compensating the deviation. These features are referred to as ***input representative features***. Second, frequency and time domain features from the output signals $Y(s)$ are used to capture the degradation level of the crossing and they are referred to as ***defect representative features***.

Using input representative features and defect representative features, the degradation level of the crossing is estimated. Both kinds of features are fed to nonlinear regressors, producing the estimated degradation level of the crossing. This estimated defect level can provide information so that maintenance engineers have more time to schedule the maintenance activities.

Based on the described problem, the proposed methodology consists of three steps: extract the transient responses and eliminate unnecessary signal (preprocessing, Section 3-2), obtain input and defect representative features (feature extraction, Section 3-3), and build the degradation model and estimate the degradation level of the crossing (Section 3-4).

3-2 Preprocessing

As discussed in Section 3-1, only parts of the signal, when a train passes over the crossing, contain information. The signal is also recorded at a high sampling frequency, which makes the size of the file on the disk considerable. Because of this condition,

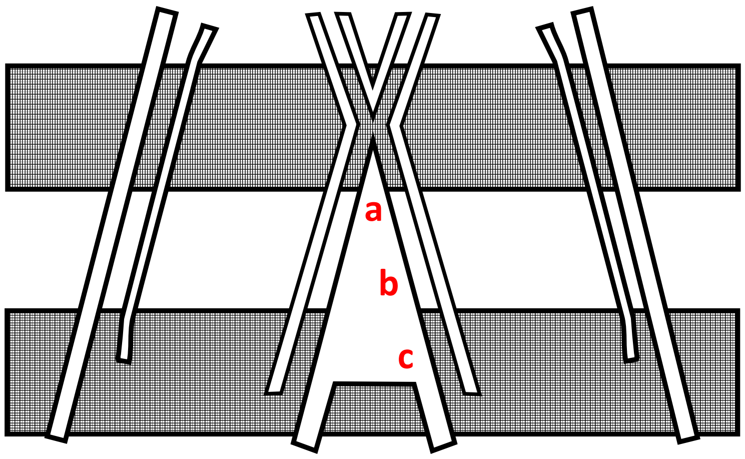


Figure 3-2: The location of accelerometers installed at the crossing.

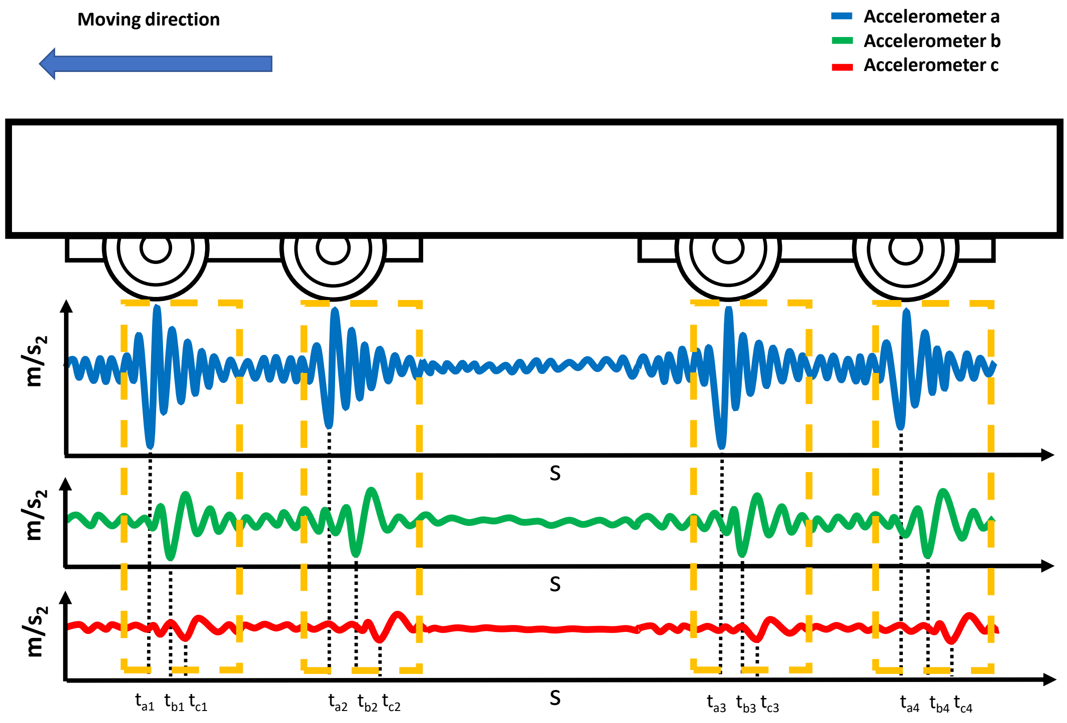


Figure 3-3: The vibration signal produced by accelerometers located in points a, b, and c.

detecting peaks induced by the wheel-crossing impact in the full acceleration signal will be inefficient and time-consuming.

Moreover, different parts of the signal representing bogies that pass over the crossing may contain peaks with significantly different magnitudes compared to the peaks from other bogies. This condition makes the wheel-crossing impact detection difficult when it is done at once over the whole signal. As the signal of a single train bogie contains peaks with similar magnitudes, the process of detecting peaks per bogie is more robust compared to detecting wheel-crossing impacts in a full acceleration signal.

In order to make the preprocessing efficient, not time-consuming, and more robust, two methods were sequentially considered: signal segmentation (Section 3-2-1), and transient responses extraction (Section 3-2-2). The purpose of the signal segmentation is to identify parts of the signal that represent bogies and eliminate parts of the signal that contains no information. The purpose of the transient responses extraction is to identify the peaks induced by the wheel-crossing impact and to extract the transient responses.

3-2-1 Signal segmentation

The segmentation is performed in two steps: dividing the acceleration signal coarsely, and error correction to eliminate undesired parts of the signal that is detected as bogies.

In the first step, the train signal is fed into an averaging filter to make a clear distinction between parts of the signal that need to be kept or eliminated [53]. The averaging filter is defined as

$$s_{\text{Avg}}(t_k) = \frac{1}{l_w} \sum_{i=-\frac{l_w-1}{2}}^{\frac{l_w-1}{2}} |s(t_k + i)| \tag{3-2}$$

where $s_{\text{Avg}}(t_k)$ is the average-filtered signal computed from the source signal $s(t_k)$, and l_w is the length of the sliding window (an odd natural number). For a simplification of the indexing in the further process, t_k refers to the index of the signal matrix (not the time stamp in second). The average-filtered signal $s_{\text{Avg}}(t_k)$ is used to distinguish which parts of the signal should be kept or eliminated by $s_{\text{m1}}(t_k)$ according to the magnitude threshold m with the following rule:

$$s_{\text{m1}}(t_k) = \begin{cases} 0, & \text{if } \max(0, [s_{\text{Avg}}(t_k - \frac{l_w-1}{2}), ..., s_{\text{Avg}}(t_k + \frac{l_w-1}{2})]) \leq m \\ 1, & \text{if } \max(0, [s_{\text{Avg}}(t_k - \frac{l_w-1}{2}), ..., s_{\text{Avg}}(t_k + \frac{l_w-1}{2})]) > m \end{cases} \tag{3-3}$$

Indices t_k of rising and dropping edges in $s_{\text{m1}}(t_k)$ are saved in arrays indicating the beginning (T_{start}) and the end (T_{end}) of the signal that need to be kept as shown in Figure 3-4. Other parts of the signal outside those indices are eliminated. In this step,

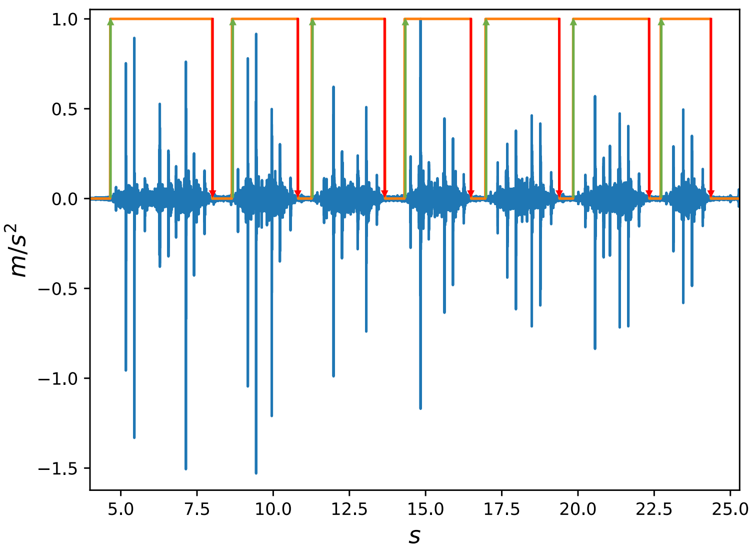


Figure 3-4: The blue signal represents the full acceleration signal. Areas under the orange lines indicate segments marked by $s_{m1}(t_k)$ using the average filter (3-2) and the threshold marker (3-3). Rising edges (green arrows upward) indicate T_{start}, and dropping edges (red arrows downward) indicate T_{end}. Magnitudes of peaks in first two segments are significantly larger than from last three segments.

the source signal $s(t_k)$ is either the full acceleration signal with its original frequency sampling or a decimated signal, depending on the frequency sampling. In the case of very high frequency sampling (e.g. 100 kHz), the signal can be decimated first to make an efficient use of the computation time without losing the important features of this part of the preprocessing.

In the second step, errors produced by the first partitioning of the signal are corrected. Short segments representing error segmentation are eliminated, and longer segments representing too coarse segmentation are repartitioned. The sum of wavelet coefficients in the time domain is also utilized to eliminate noises on the segments, as the wavelet denoising can remove a considerable amount of noises while preserving the sharp features in the signal [54]. The sum of wavelet coefficients in the frequency domain $C_f(t_k)$ that is defined as

$$C_f(t_k) = \sum_{f=f_0}^{f_t} C(f, t_k) \tag{3-4}$$

Where f_0 is the lower bound frequency of the wavelet transform, f_t is the upper bound frequency of the wavelet transform, and $C(f, t_k)$ is the result of CWT on $s(t_k)$, will show more significant magnitudes in the area of transient responses. The time domain t in the result of Continuous Wavelet Transform (CWT) is adapted to the matrix index t_k

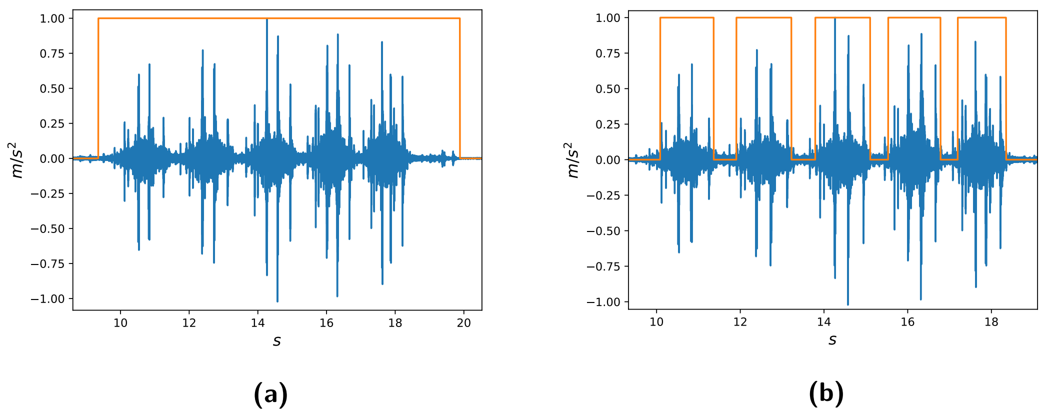


Figure 3-5: The example of the failed segmentation performed by $s_{\mathrm{m1}}(t_k)$ (a), where the blue signal is the full acceleration signal, and parts of the signal under the orange line are divided segments. After the segmentation is corrected using $s_{\mathrm{m2}}(t_k)$, the full acceleration signal was properly segmented (b).

for a simplification of the further processing. The error correction is perform with the following rule:

$$s_{\mathrm{m2}}(t_k) = \begin{cases} 0, & \text{if } T_{\mathrm{end}} - T_{\mathrm{start}} \leq p \\ 1, & \text{if } C_f(t_k) > q, \forall t_k \in [T_{\mathrm{start}}, T_{\mathrm{end}}] \end{cases} \tag{3-5}$$

where p and q are predefined constants. Indices t_k of rising and dropping edges in $s_{\mathrm{m2}}(t_k)$ are saved in arrays indicating the beginning (T_{start}) and the end (T_{end}) of the signal that need to be kept. Results of both steps of the segmentation are shown in Figures 3-5 (a) and (b).

3-2-2 Transient responses extraction

After obtaining the segments of signals with the effect of passing bogies, the next steps are to detect the locations of the correct peaks and extract the transient responses located around them. Not all peaks in the segments represent wheel impacts. Some of them are caused by the welds before and after the crossing [23]. The following properties are considered to identify which peaks should be extracted from the segments:

- When middle coaches are passing over the crossing, two near bogies can be detected as a segment with four peaks representing wheels. A single bogie is always detected as a segment with two peaks representing wheels.
- The first and the last segment of the signal can correspond to one bogie (electrical module unit), or three closely-located bogies (two bogies from the locomotive and one bogie from a passenger coach).

- Impulses induced by wheel-crossing impacts are higher than the impulses induced by other events such as welds.

- Impulses induced by the wheel impacts decay slower than the other peaks.

Considering these properties of the signal, wheel impacts can be detected as several dominant peaks in the segment. The number of the dominant peak is determined according to the position of the segment in the signal.

In the opening segment and the closing segment, the number of dominant peaks can be two or six. In the case of more than two closely-located bogies, the length of the segment will be larger than the average length of middle segments. Thus, six wheel impacts, which come from two bogies of a locomotive and a bogie of a passenger coach, must be detected. Otherwise, two wheel impacts must be detected.

In the middle segment, the number of dominant peaks can be two or four. Light trains can have two dominant peaks as there is only one bogie located between two coaches. However, most trains have two closely-located bogies in the middle segment of the signal. In this case, four dominant peaks must be detected. The number of dominant peaks that must be detected can be decided based on the difference of the length of the opening segment and the average length of middle segments. As the opening segment of light trains always contains one bogie, the length of the opening segment should not have much difference compared to the average length of middle segments. In this case, two dominant peaks must be detected. Otherwise, four dominant peaks must be detected.

Before detecting peaks, the segment is smoothened using the wavelet denoising [54], which is defined as C_f in (3-4), to eliminate noises in the segment. This algorithm is expressed in Algorithm 1, which calls Algorithm 2 by function $DETECT\_N\_PEAKS(\cdot)$ to find indices of a particular number of peaks in a particular segment.

For the detection of peaks in a segment in Algorithm 2, positions of the peaks are obtained one by one starting from the highest magnitude. Once the first location of the peak has been retrieved, parts of the signal related to the detected peak are flattened to their minimum level. This is done by finding two indices with the local minimum located before and after the peak, and change the magnitude of the signal between the indices into its minimum value. The process is then iterated. This way, the indices are sorted according to their magnitudes. Indices produced by Algorithm 1 and Algorithm 2 (*indices*) should be upsampled to generate predicted indices with the original frequency sampling, in the case of the signal segmented was downsampled. The real indices of wheel impacts are then found by looking at the minimum magnitudes near the predicted indices (the wheel impacts apply downward forces; thus, detected peaks are always negative). The result of detected peaks are shown in Figure 3-6.

Algorithm 1 Find the correct positions of transient responses in the segment.

```
1: $S$ :  array of the acceleration signal (or downsampled acceleration signal) over the
   whole measurement period
2: $T_{start}$ :  array of the start index of the segment
3: $T_{end}$ :  array of the end index of the segment
4: $T_{end}$ :  predefined real numbers
5: procedure ObtainIndices($S, T_{start}, T_{end}$)
6:     $n\_seg \leftarrow LENGTH(T_{start})$
7:     $avg\_segment\_len \leftarrow \text{average}(T_{end}[2 : n\_seg - 1] - T_{start}[2 : n\_seg - 1])$
8:     for $i = [1, \ldots, n\_seg]$ do
9:         $segment[i] \leftarrow C_f(S[T_{start}[i] : T_{end}[i]])$
10:        $segment\_len[i] \leftarrow T_{end}[i] - T_{start}[i]$
11:        if $i = 0$ then
12:            if $segment\_len[i] < avg\_segment\_len$ then
13:                $indices[i] \leftarrow DETECT\_N\_PEAKS(segment[i], 2)$
14:            else
15:                $indices[i] \leftarrow DETECT\_N\_PEAKS(segment[i], 6)$
16:            end if
17:        else if $i < n\_seg$ then
18:            if $(|segment\_len[0] - avg\_segment\_len|/avg\_segment\_len) <$
   $const\_length$ then
19:                $indices[i] \leftarrow DETECT\_N\_PEAKS(segment[i], 2)$
20:            else
21:                $indices[i] \leftarrow DETECT\_N\_PEAKS(segment[i], 4)$
22:            end if
23:        else
24:            if $segment\_len[i] < avg\_segment\_len$ then
25:                $indices[i] \leftarrow DETECT\_N\_PEAKS(segment[i], 2)$
26:            else
27:                $indices[i] \leftarrow DETECT\_N\_PEAKS(segment[i], 6)$
28:            end if
29:        end if
30:    end for
31:    Return $indices$
32: end procedure
```

Algorithm 2 Detect dominant peaks in the segments.

```
1: $s$ :  one segment of the acceleration signal
2: $n$ :  the number of peaks that need to be found
3: procedure DETECT_N_PEAKS($s, n$)
4:     for $i = [1, \dots, n]$ do
5:         $t_k \leftarrow \text{argmax}(s)$
6:         $indices[i] \leftarrow t_k$
7:         $break\_low \leftarrow False$
8:         $break\_up \leftarrow False$
9:         $p \leftarrow$ any small natural number
10:        $q \leftarrow$ any small natural number
11:        loop
12:            if $break\_low = False$ then
13:                $low\_index = \text{argmin}(s[t_k - p : t_k])$
14:            end if
15:            if $break\_up = False$ then
16:                $up\_index = \text{argmin}(s[t_k : t_k + q])$
17:            end if
18:            if $low\_index > t_k - p$ then
19:                $break\_low = True$
20:            else
21:                $p \leftarrow p + 1$
22:            end if
23:            if $up\_index < t_k + q$ then
24:                $break\_up = True$
25:            else
26:                $q \leftarrow q + 1$
27:            end if
28:            if ($break\_low = True$) and ($break\_up = True$) then
29:                Break loop
30:            end if
31:        end loop
32:        $s[t_k - low\_index : t_k + up\_index] = \min(s[t_k - low\_index : t_k + up\_index])$
33:    end for
34:    Return $indices$
35: end procedure
```

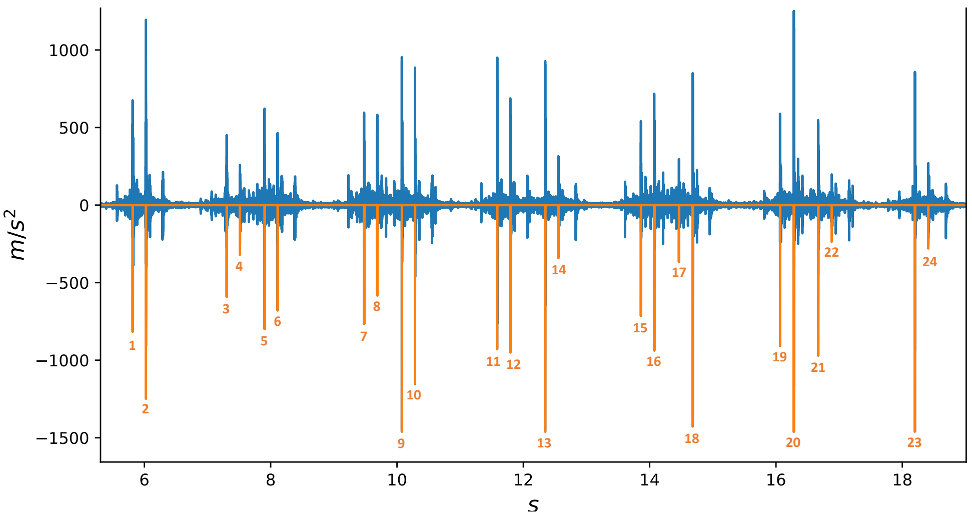


Figure 3-6: The blue signal indicates the full acceleration signal and the orange signal indicates the detected wheel impacts. Elements of the blue signal with same indices as the indices of detected wheel impacts are copied to the null matrix to create the plot of the orange signal.

3-3 Feature extraction

Two kinds of features should be extracted from the transient responses as illustrated in Figure 3-7: input representative features (Section 3-3-1) and defect representative features (Section 3-3-2). As discussed in Section 3-1, both features affect the transient responses. Thus, it is difficult to separate one feature from another. In order to solve this problem, learning algorithms are utilized.

3-3-1 Input representative features

Input representative features are binary values in a column vector representing the characteristics of the input [29]. Since the operation in a multi-layer perceptron can be described as a multiplication between an input vector and the weight matrix, a binary input vector works as an activator of a particular mode of weight. For example, a record that has input representative features in a column vector $X^{l-1} = [0\ 0\ 0\ 1]^T$ will activate the last column vector of weight matrix W^l in

$$X^l = \max(0, W^l X^{l-1} + B^l) \tag{3-6}$$

where l indicates the current layer, and $l-1$ indicates the previous layer. In the case of railway crossings, the type of the train can be utilized as an input representative feature. The premise is that trains of the same type have similar dynamical responses.

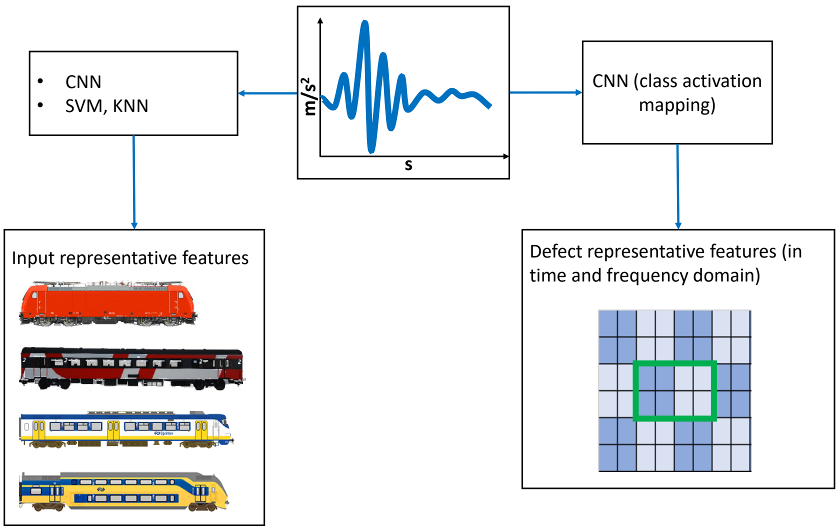

CNN
SVM, KNN
m/s²
s
CNN (class activation mapping)
Input representative features
Defect representative features (in time and frequency domain)

Figure 3-7: Features extracted from the transient responses: the input representative features, and the defect representative features.

Moreover, they also use the same kind of bogies. It is very relevant to assume that the wheel characteristics from trains of the same type are also similar.

In this thesis, types of trains were obtained using CNN, Support Vector Machine (SVM), and K-Nearest Neighbors (K-NN). SVM and K-NN were chosen because they have been widely used as a benchmarking standard for similar tasks, e.g. in [44, 55, 56]. Various types of input extracted from transient responses were chosen depending on the characteristics of the classifier. For CNN classifiers, wavelet coefficients were utilized. For SVM and K-NN, statistical values derived from the transient responses and wavelet time-frequency entropy extracted from the wavelet coefficients were used. Statistical values utilized as the input are max, average, variance, standard deviation, median, the sum of area under the curve, distances between peaks of accelerometers, and the time constant of the signal. Statistical features were also used in [44], resulting an overall accuracy of 90.7% using SVM classifier. In the case of wavelet time entropy, the dimension of wavelet coefficients is reduced into a vector in time domain by means of sum and averaging as

$$w_t(i) = \frac{1}{l_s} \sum_{t=1+l_s(i-1)}^{l_s i} \sum_{f=f_0}^{f_t} C(f,t) \tag{3-7}$$

$$l_s = \frac{n_{tr}}{m} \tag{3-8}$$

for all $w_t \in W_t$, where $W_t \in \mathbb{R}^m$ denotes the wavelet time entropy, $i \in [1, \ldots, m]$ is the index of W_t, n_{tr} is the length of transient responses, f_0 is the lower bound frequency of the wavelet transform, f_t is the upper bound frequency of the wavelet transform. Mathematically speaking, the wavelet time entropy is the reduction of the wavelet coefficients, which is performed by summing the matrix in the frequency domain f. The result of the summation is then divided into m number of regions. In each region, with the size l_s, a representative value is obtain using an average.

Two different CNN architectures are proposed in this thesis as illustrated in Figure 3-8 and Figure 3-9. The architecture in Figure 3-8 uses different convolutional networks for each accelerometer and the architecture in Figure 3-9 uses one convolutional network for all accelerometers. Both configurations have different level of flexibility. When one of the accelerometer contains noises, eparated convolutional networks are preferred. Both architectures, separated and shared convolutional networks can be combined to arrange the flexibility of the convolutional networks. SVM and K-NN are explained in Appendix A-1 and A-2 respectively.

3-3-2 Defect representative features

Defect representative features can be obtained using the class activation mapping [57, 58] applied on the wavelet coefficients. This method could indicate the area of wavelet coefficients that significantly contributed to the output of the CNN algorithm, which is time and frequency of the vibration signal indicating degradation of the crossing. Class activation mapping works with a particular architecture shown in Figure 3-10. Suppose the output of a CNN architecture is defined as follows:

$$y = W^T X \tag{3-9}$$

where y denotes the output value, $W \in \mathbb{R}^{pqr}$ denote the weight vector, and $X \in \mathbb{R}^{pqr}$ represents the input vector. Since X is obtained by the vectorization of last feature maps:

$$X = \text{vec}(A) \tag{3-10}$$

where $A \in \mathbb{R}^{p \times q \times r}$, and a matrix $\hat{W} \in \mathbb{R}^{p \times q \times r}$ can also be constructed by

$$\hat{W} = \text{vec}^{-1}(W) \tag{3-11}$$

where function $\text{vec}^{-1}(.)$ converts a vector of $\mathbb{R}^{pqr}$ into a matrix $\mathbb{R}^{p \times q \times r}$. Another matrix $\bar{W} \in \mathbb{R}^{s \times t \times r}$ is then expanded from the $\hat{W} \in \mathbb{R}^{p \times q \times r}$ with the following rule:

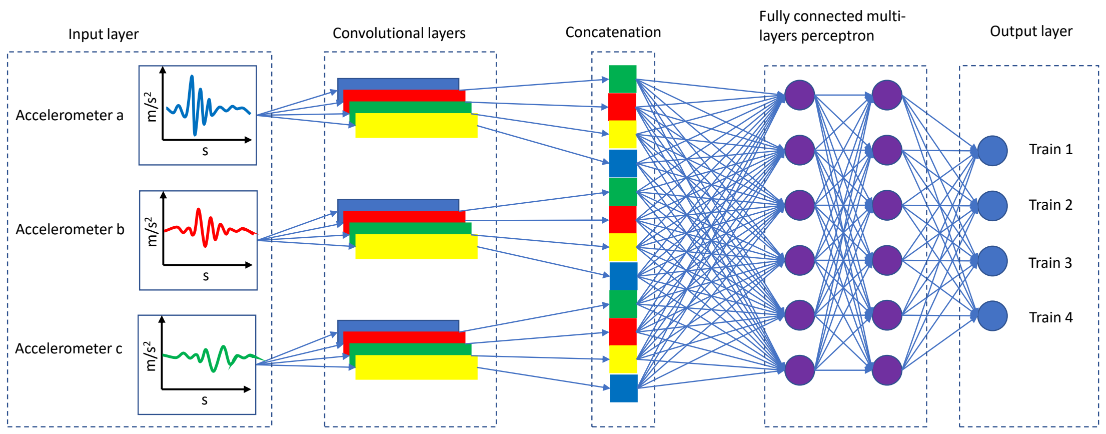


Figure 3-8: A CNN architecture with separated convolutional networks per input signal.

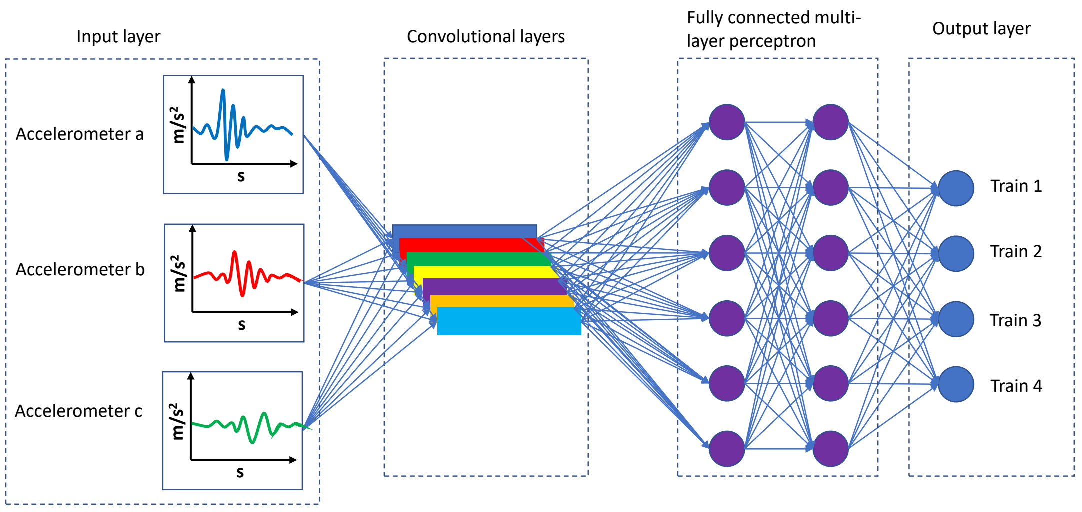


Figure 3-9: A CNN architecture with shared convolutional networks per input signal.

$$\bar{w}_{mnk} = \hat{w}_{ijk} \tag{3-12}$$

$$i = \left\lceil \frac{m}{(r/p)} \right\rceil \tag{3-13}$$

$$j = \left\lceil \frac{n}{(s/q)} \right\rceil \tag{3-14}$$

for all $m \in [1, \ldots, s]$, $n \in [1, \ldots, t]$, $i \in [1, \ldots, p]$, $j \in [1, \ldots, q]$, $k \in [1, \ldots, r]$, $\bar{w}_{mnk} \in \bar{w}$, and $\hat{w}_{ijk} \in \hat{w}$. The localization map L_{CAM} is defined as:

$$L_{\text{CAM}} = \sum_{k=1}^{r} \bar{w}_{mnk} c_{mnk}, \tag{3-15}$$

for all $c_{mnk} \in C$, and $\bar{w}_{mnk} \in \bar{W}$, where $C \in \mathbb{R}^{s \times t \times r}$ is the last feature maps. Mathematically speaking, the L_{CAM} is the weighted summation of last feature maps. Each weight in the fully connected layer receives the input from last pooling maps, where each element of pooling maps represents a particular area of last feature maps. These weights, which are back propagated to last feature maps, can be used to amplify areas of feature maps that contribute positively to the output and suppressed areas that contribute negatively to the output. By setting the proper areas of interests, the effect of features in the defined areas can be seen from the visualization of the L_{CAM}. The problem can arise when the resolution of the areas is set too high, which makes the size of the weights grow excessively. This condition can lead to inaccurate visualization of the L_{CAM} due to the weights are not trained well, especially when the number of training sample is not sufficient.

In this thesis, the wavelet coefficients were divided into three different areas of interests, as illustrated in Figure 3-11. The purpose of this arrangement is to find out in which domain of the wavelet coefficient matrix contains features related to the defects. If the features are located in a particular time domain, the arrangement that is shown in Figure 3-11 (a) will give the area that is highlighted constantly when defects exist. If the features are located in a particular frequency domain, the arrangement that is shown in Figure 3-11 (c) will give the area that is highlighted constantly when defects exist. If defects can be inspected from both time and frequency domain, the arrangement that is shown in Figure 3-11 (b) will give the area that is highlighted constantly when defects exist.

In order to produce an accurate L_{CAM} visualization, the trend closely related to the defects must be utilized as a baseline value for the regression test. As discussed in [13], the number of passing axles are closely related to the defects as in the simulation they could be used to predict the initiation of the surface defects on the crossing. Thus, wavelet coefficients can also be utilized in regression tests to estimate the number of passing axles, in order to produce the visualization map. Moreover, different length of

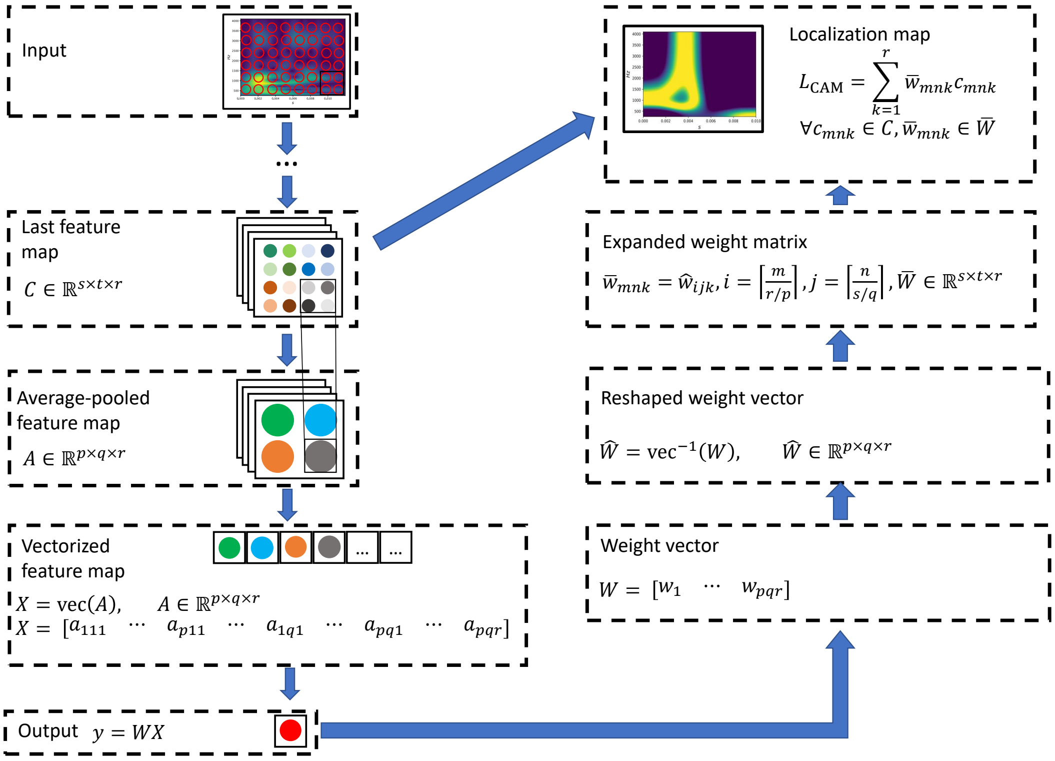

Input
...
Last feature map
$C \in \mathbb{R}^{s\times t\times r}$
Average-pooled feature map
$A \in \mathbb{R}^{p\times q\times r}$
Vectorized feature map
...
...
$X = \mathrm{vec}(A), \quad A \in \mathbb{R}^{p\times q\times r}$
$X = [a_{111} \; \cdots \; a_{p11} \; \cdots \; a_{1q1} \; \cdots \; a_{pq1} \; \cdots \; a_{pqr}]$
Output $y = WX$
Localization map
$L_{\mathrm{CAM}} = \sum_{k=1}^{r} \bar{w}_{mnk} c_{mnk}$
$\forall c_{mnk} \in C, \bar{w}_{mnk} \in \bar{W}$
Expanded weight matrix
$\bar{w}_{mnk} = \hat{w}_{ijk}, i = \left\lceil \frac{m}{r/p} \right\rceil, j = \left\lceil \frac{n}{s/q} \right\rceil, \bar{W} \in \mathbb{R}^{s\times t\times r}$
Reshaped weight vector
$\hat{W} = \mathrm{vec}^{-1}(W), \quad \hat{W} \in \mathbb{R}^{p\times q\times r}$
Weight vector
$W = [w_1 \; \cdots \; w_{pqr}]$

Figure 3-10: The CNN used for the class activation mapping.

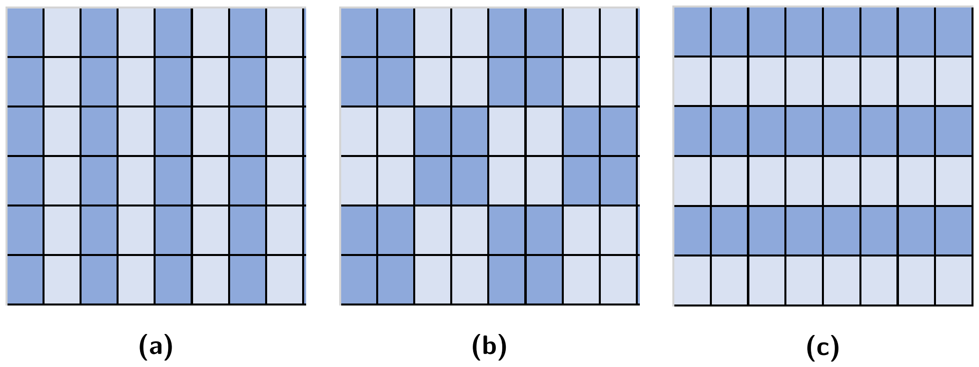


Figure 3-11: The wavelet coefficient matrices are divided into three areas of interests (frequency $\times$ time): (a) 1×8 focusing on the time domain, (b) 3×4 focusing on both the frequency and time domain, and (c) 6×1 focusing on the frequency domain. Each area of interests is indicated by two different colors (the dark blue and the light blue).

transient responses as shown in Figure 3-12 are also considered as the longer responses contain more information such as time constants of the transient responses.

3-4 Estimating health conditions of the crossing

The last step is to estimate the health condition of the crossing based on the input representative features and defect representative features. This step is illustrated in Figures 3-13. Four CNN architectures are considered to perform this task as shown in Figure 3-14 to 3-17. In the model shown in Figure 3-14, the architecture consists of five convolutional networks arranged in series. Each network reduces the dimension of the input (mainly in the time domain) using a shared weight convolutional layer shown by orange boxes in Figure 3-14. Unlike the max or average-pooling layer, a shared weight convolutional layer allows more flexible mathematical operation for representing the data, other than taking the maximum or average value. This model is similar to VGG Net without pooling layers in [59, 49], and referred to **ConvNet** in this thesis. In other models shown in Figure 3-16 to 3-17, the architectures consist of more complicated layers compared to ConvNet. Each layer contains convolutional networks and pooling layers processing the same data in parallel to add different pieces of information. At the end of each layer, the output of convolutional and pooling networks are concatenated shown by violet boxes in Figure 3-16 to 3-17. This kind of architecture is referred to **Inception**, and has been used in [60] to increase the accuracy.

3-5 Summary

This chapter has discussed the methodology proposed in this thesis: the problem description that needs to be tackled, and steps of the methodology designed for solving

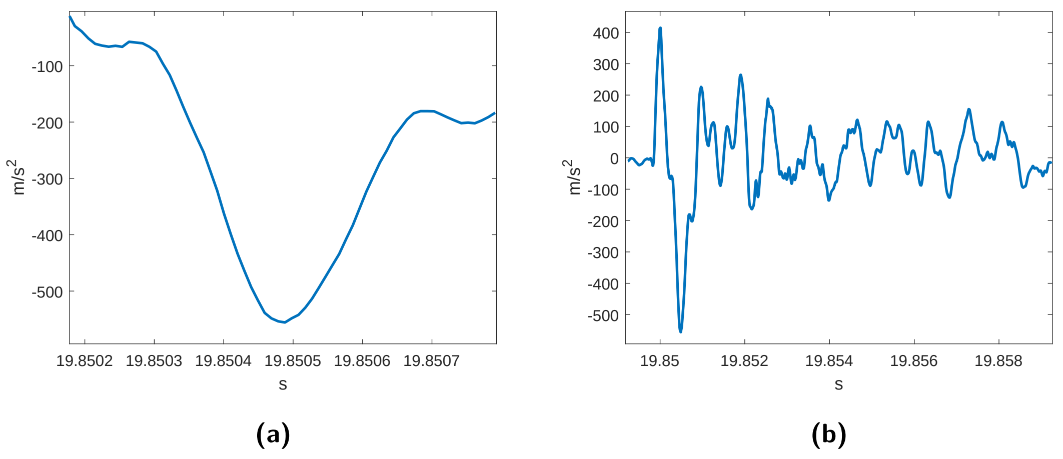


Figure 3-12: Different lengths of transient responses used for the activation mappings: (a) $0.625\,\mathrm{ms}$ (only the peak) and (b) $10\,\mathrm{ms}$ (a full transient response).

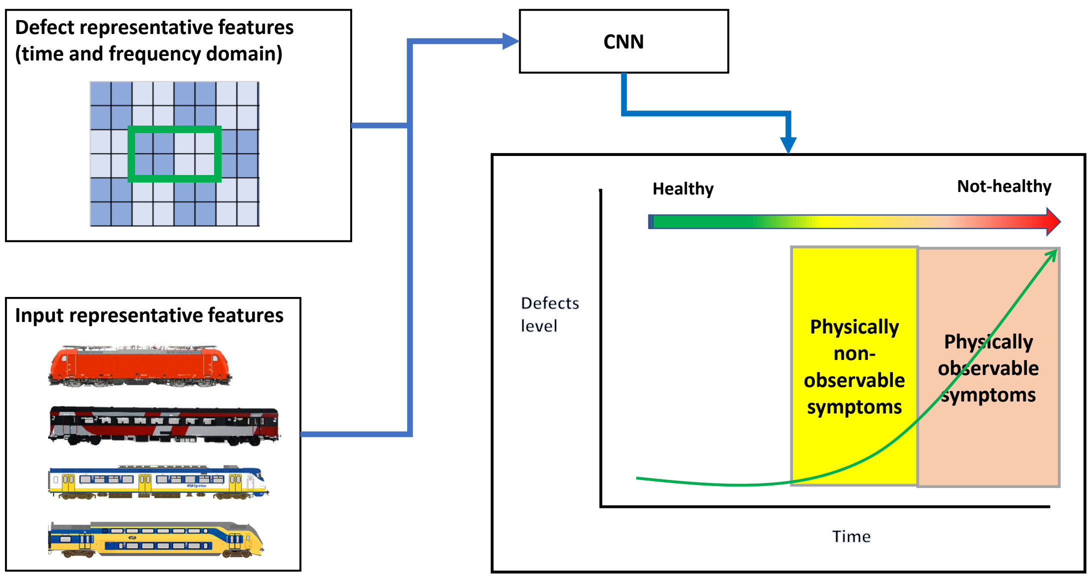

Defect representative features
(time and frequency domain)
CNN
Input representative features
Healthy
Not-healthy
Defects level
Physically non-observable symptoms
Physically observable symptoms
Time

Figure 3-13: The input representative features and defect representative features are fed to the nonlinear regressor to estimate the defects level in the crossing.

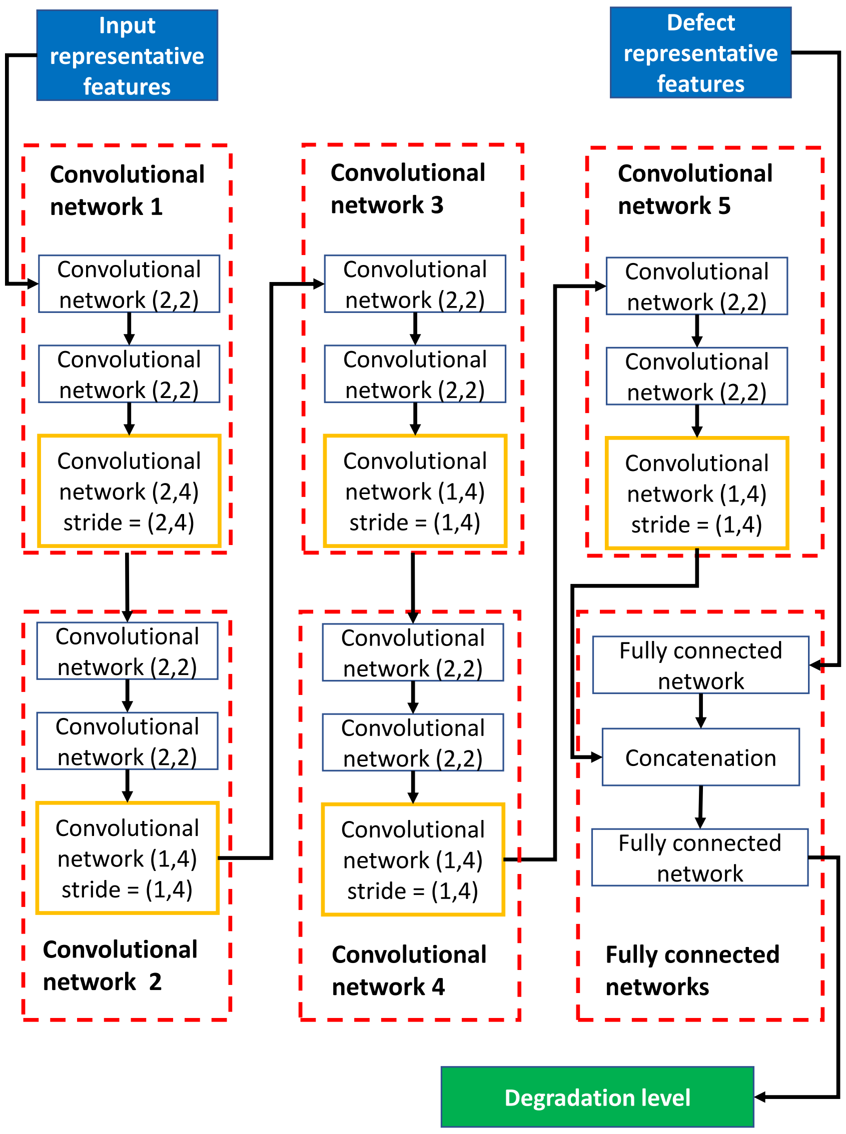

Input representative features
Defect representative features
Convolutional network 1
Convolutional network (2,2)
Convolutional network (2,2)
Convolutional network (2,4) stride = (2,4)
Convolutional network (2,2)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Convolutional network 2
Convolutional network 3
Convolutional network (2,2)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Convolutional network (2,2)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Convolutional network 4
Convolutional network 5
Convolutional network (2,2)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Fully connected network
Concatenation
Fully connected network
Fully connected networks
Degradation level

Figure 3-14: CNN architecture with fully convolutional networks (ConvNet).

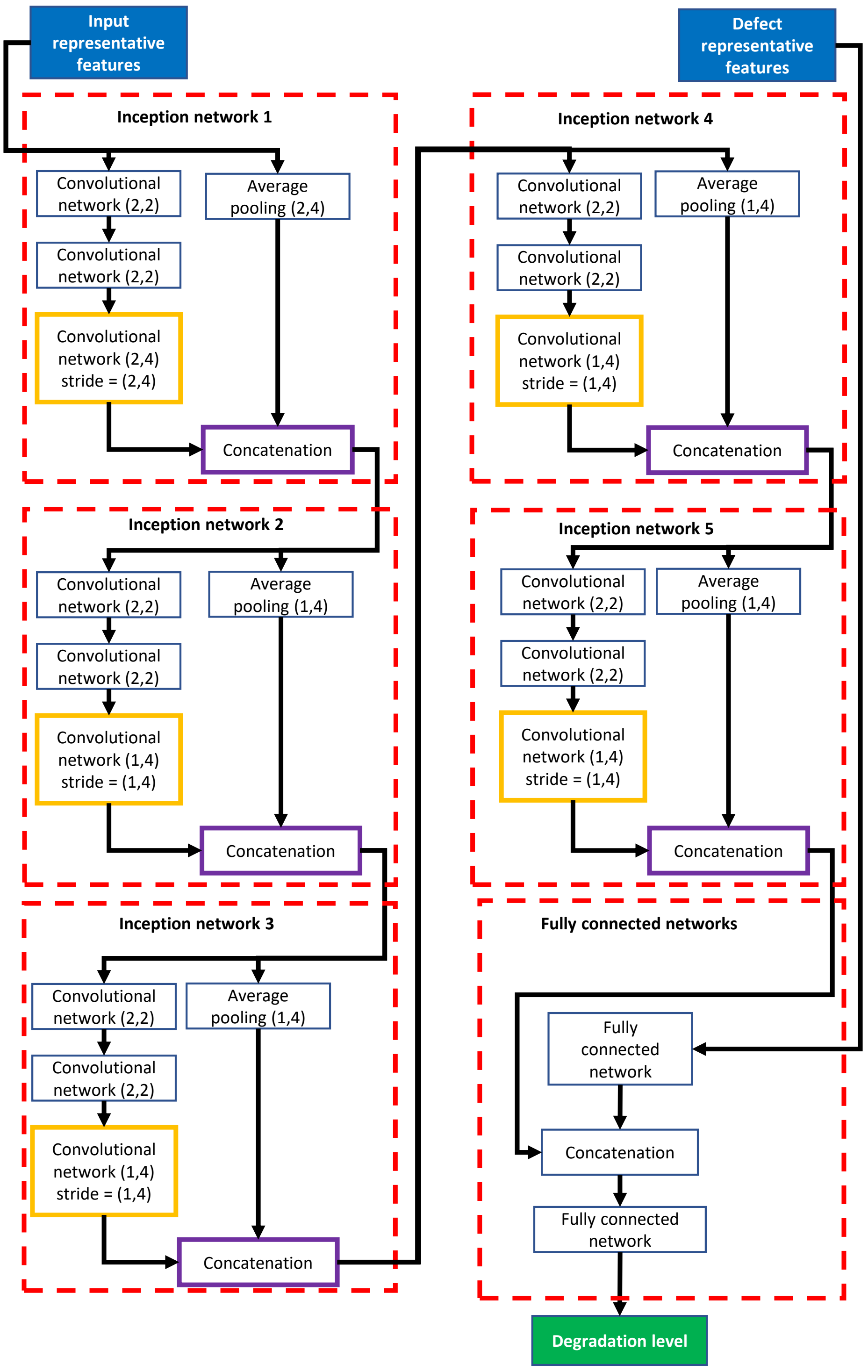

Input representative features
Defect representative features
Inception network 1
Convolutional network (2,2)
Average pooling (2,4)
Convolutional network (2,2)
Convolutional network (2,4) stride = (2,4)
Concatenation
Inception network 2
Convolutional network (2,2)
Average pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 3
Convolutional network (2,2)
Average pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 4
Convolutional network (2,2)
Average pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 5
Convolutional network (2,2)
Average pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Fully connected networks
Fully connected network
Concatenation
Fully connected network
Degradation level

Figure 3-15: A CNN architecture with inception networks consisting of convolutional and average-pooling layers arranged in parallel (Inception Conv + Avg).

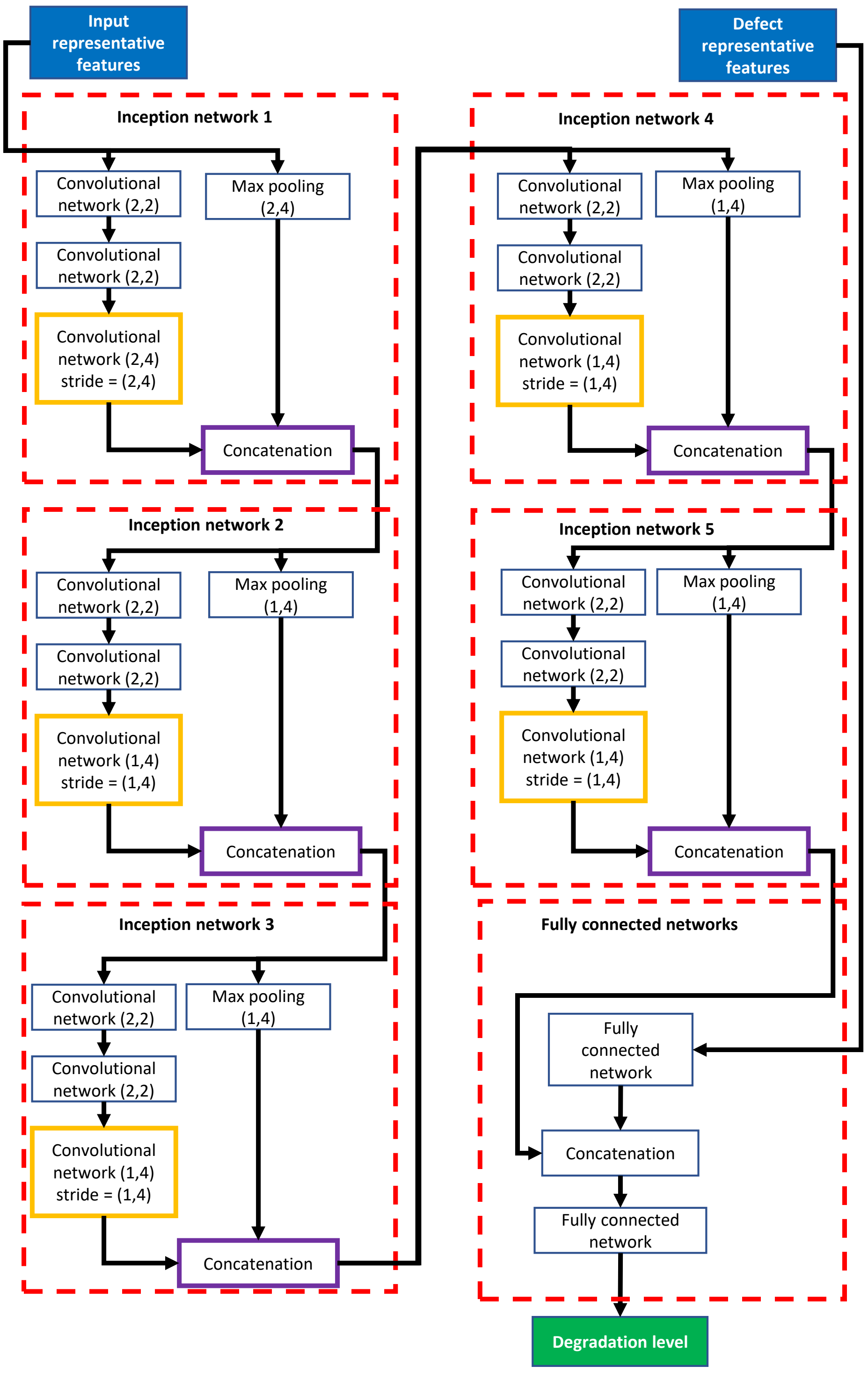

Input representative features
Defect representative features
Inception network 1
Convolutional network (2,2)
Max pooling (2,4)
Convolutional network (2,2)
Convolutional network (2,4) stride = (2,4)
Concatenation
Inception network 2
Convolutional network (2,2)
Max pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 3
Convolutional network (2,2)
Max pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 4
Convolutional network (2,2)
Max pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 5
Convolutional network (2,2)
Max pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Fully connected networks
Fully connected network
Concatenation
Fully connected network
Degradation level

Figure 3-16: A CNN architecture with inception networks consisting of convolutional and max-pooling layers arranged in parallel (Inception Conv + Max).

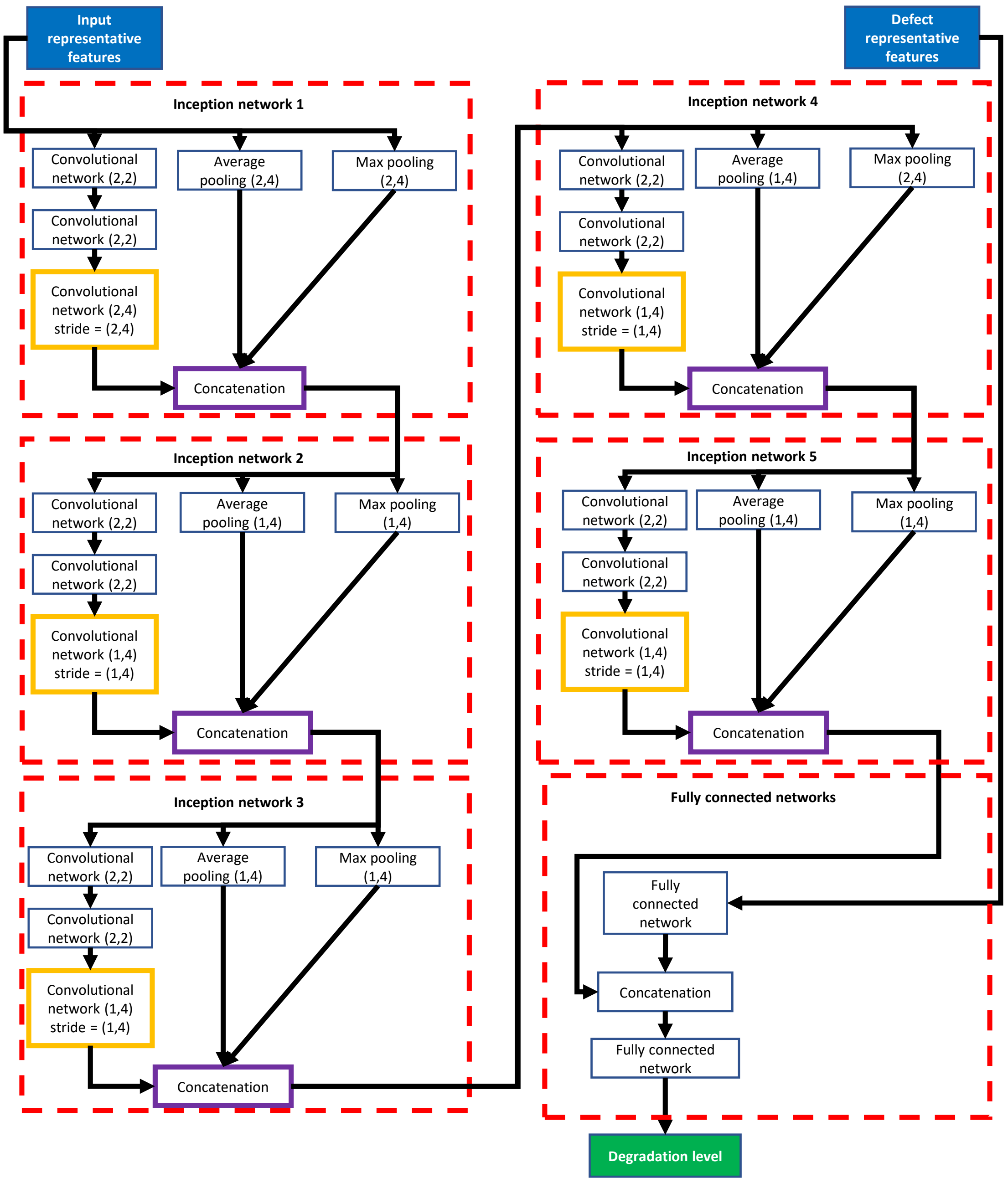

Input representative features
Defect representative features
Inception network 1
Convolutional network (2,2)
Average pooling (2,4)
Max pooling (2,4)
Convolutional network (2,2)
Convolutional network (2,4) stride = (2,4)
Concatenation
Inception network 2
Convolutional network (2,2)
Average pooling (1,4)
Max pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 3
Convolutional network (2,2)
Average pooling (1,4)
Max pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 4
Convolutional network (2,2)
Average pooling (1,4)
Max pooling (2,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Inception network 5
Convolutional network (2,2)
Average pooling (1,4)
Max pooling (1,4)
Convolutional network (2,2)
Convolutional network (1,4) stride = (1,4)
Concatenation
Fully connected networks
Fully connected network
Concatenation
Fully connected network
Degradation level

Figure 3-17: A CNN architecture with inception networks consisting of convolutional, average-pooling, and max-pooling layers arranged in parallel (Inception Conv + Avg + Max).

the problem.

The proposed methodology uses responses of the wheel-crossing impact, which is recorded by several accelerometers installed on the crossing. These responses are affected by both defects on the crossing and the wheel-crossing impact. As wheel-crossing impacts are induced by various trains from the different type, the responses recorded by accelerometers become. Because of this condition, estimating the degradation level of the crossing is difficult to perform analytically.

Different problems also arise from the acceleration data induced by the wheel-crossing impact. In order to capture transient responses, a high frequency sampling must be used by the accelerometers. As a consequence, the size of the data is considerable. Additionally, only particular parts of the signal contain information. Thus, processing the full acceleration signal is inefficient and computationally expensive.

Three steps of the methodology are designed for solving the problem regarding the variation of the input caused by different types of trains, and the size of the data caused by a high frequency sampling: preprocessing, feature extraction, and estimation of the health condition of the crossing.

In the preprocessing, parts of the signal containing wheel-crossing impacts are extracted from the full acceleration signal. In order to make the process more efficient, the acceleration signal is segmented into several parts. This segmentation is performed according to the threshold values that are applied to the output of the averaging filter [53] and the output of the wavelet denoising filter [54]. After the acceleration signal is segmented, indices of wheel-crossing impacts are detected, so that the transient responses can be extracted.

Transient responses that are extracted from the acceleration data are utilized to build a degradation model of the crossing. In order to build the model, wavelet coefficients are computed from the responses using CWT. These wavelet coefficients are fed to a particular type of CNN regressor to construct the localization map using the class activation mapping [57, 58]. The localization map is the visualization of the weighted sum of last feature maps. As the weights are back propagated from the fully connected layer, the localization map shows the area of the input (i.e. wavelet coefficients) that has the most contribution to the output. When a trend related to defects, such as the number of passing axles [13], are utilized in the regression test to build the localization map, the location of features that indicate the defects can be approximated. In this thesis, these features are called the defect representative features. Using the defect representative features, the degradation level of the crossing can be built.

The degradation model built from defect representative features contains high deviations. Therefore, compensating features are needed to suppress these deviations. These features are binary features [29], and they are referred to input representative features. As deviations on the degradation model are caused by various inputs, which are related to trains of the different type that induce the responses, types of trains that pass the crossing are used as input representative features. In order to extract the type of the train from the wavelet coefficient, different kinds of CNN classifiers are utilized. The CNN classifier with separated convolutional networks per sensor, and the CNN classifier with shared convolutional networks for all sensors [29, 47]. CNN classifiers are chosen because they are good for performing classification tasks from high dimensional input data such as wavelet coefficients. For comparison, machine learning classifiers such as SVM and K-NN are also used to extract the features. These machine learning classifiers have been used as benchmarks for similar classification tasks [44, 55, 56].

With input representative features and defects representative features, several degradation models with fewer deviations are built using CNN regressors with various configurations. Two main configurations can be applied to the CNN regressors: the fully connected convolutional networks [59, 49], and the inception networks [60]. The inception networks are extended into three configurations: inception networks with the average-pooling layers, inception networks with max-pooling layers, and inception networks with both average-pooling layers and max-pooling layers.

Chapter 4

Performance Evaluation

As discussed in Chapter 3, the proposed methodology consists of three steps: preprocessing for extracting transient responses from the signals, feature extraction for obtaining indicating and compensating features from the transient responses, and estimating the degradation level of the crossing. The proposed methodology was implemented in Python Keras [61], and evaluated using a case study in Amsterdam.

4-1 The case study of a railway crossing in Amsterdam

The Section of Railway Engineering from TU Delft performed measurements of a crossing located behind ProRail West office, near Nieuwe Westerdokstraat, Amsterdam. The location is shown in Figure 4-1. Acceleration signals induced by the passing trains were recorded. The measurements were done from February 2015 to April 2016. Small surface defects were found during an inspection in August 2015, and the crossing was repaired in November 2015.

A prototype consisting of an STM-32 module and four 603C00 IMI accelerometers was developed as a data logger. The module sampled the signals produced by the accelerometers at 102400 Hz, allowing the frequency analysis to perform up to 10240 Hz, five times higher than the Nyquist rate required to avoid the aliasing error. The prototype captured the signals produced by four accelerometers, then saved the data into a Matlab compatible file per one passing train. The information flow is illustrated in Figure 4-2.

4-1-1 Sensor arrangement

The accelerometers were installed in different locations of the crossing as illustrated in Figure 4-3 a, and produced different signals as shown in Figure 4-3 b. The Accelerometer

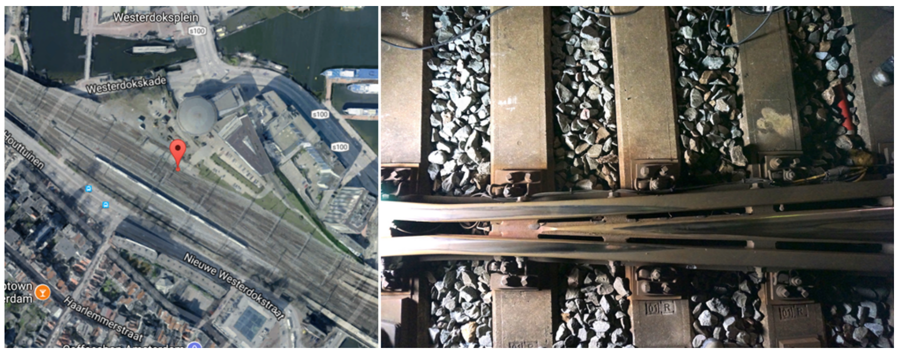


Figure 4-1: The location of case study performed by the Section of Railway Engineering from TU Delft (left) and the measured crossing (right).

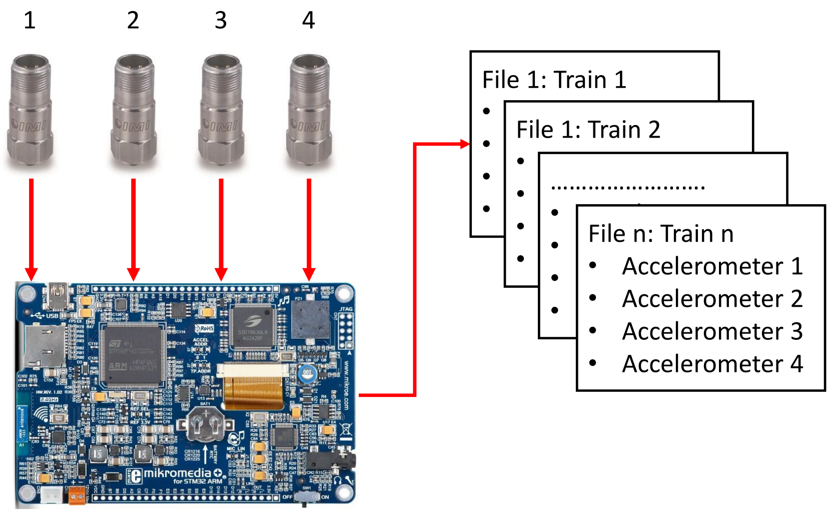
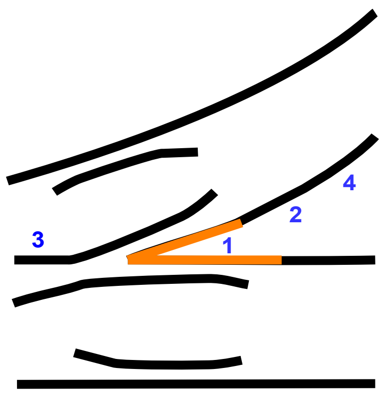
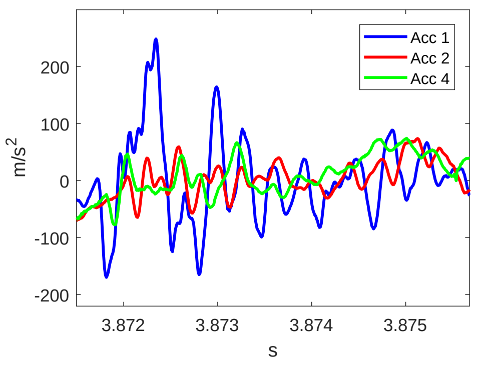


Figure 4-2: A prototype consists of four 603C00 IMI accelerometers and an STM-32 module. All measurement data of a passing train was saved by the prototype.



(a)



(b)

Figure 4-3: Arrangement of accelerometers used in the case study: (a) locations and (b) produced signals.

1 produced signals with the highest magnitude as it was located near the crossing nose, where the strongest wheel-crossing impact happens, followed by Accelerometer 2 and Accelerometer 4. Accelerometer 3, which was used to compare dynamic responses at different locations, produced the smallest magnitude of signals as it was further away from the location of wheel-crossing impact compared to the others.

4-1-2 Dataset

This case study recorded responses induced by four different types of trains: TRAXX, ICRm, SGMm, SLT, Thalys, and VIRM. In this thesis, only four of them are used: TRAXX, ICRm, SGMm, and VIRM. Thalys trains and SLT trains are not used because the number of responses induced by both trains are insufficient for training and evaluation.

The TRAXX is a type of locomotives. A TRAXX locomotive has length, width, and height of 18.90 m, 2.97 m, and 4.98 m respectively. The locomotive has the weight of is 84 mt and the speed of up to 140 km/h on the normal track, and 160 km/h on the high-speed track.

The ICRM is a type of passenger coaches. This type of coach has the length, width, and height are 26.4 m, 2.82 m, and 3.93 m. An ICRM coach also has the mass of 41 mt. In the dataset, ICRM coaches are always pulled by a TRAXX locomotive. Both TRAXX and ICRm, together with their induced vibrations, are illustrated in Figure 4-4.

The SGMm is a type of rapid transit trains. This type of trains have two different series: SGM-II and SGM-III [62]. The length, width, and height of an SGM-II train are 52.2 m, 2.8 m, and 3.89 m respectively. A train with this series has the mass of 104.2 mt and can achieve 120 km/h of speed. An SGM-III train can reach the same speed as an SGM-II train. However, the SGM-III train has a longer body (78.7 m), and a greater total mass (144 mt). There is no information regarding the series of SGMm trains in the dataset. Therefore, both of train series is assumed as one type. The first two cars of SGM-III train, and their induced vibrations, are illustrated in Figure 4-5.

The VIRM is a type of double-decker intercity trains. The VIRM train has two formations of the cars: 4 and 6. A four-cars VIRM has 235.7 mt of weight and 106.8 m of length [63]. A six-cars VIRM has 349 mt of weight, and 162.1 m of length. Although in service the VIRM trains run at 140 km/h, they are designed to achieve 200 km/h of speed. The first two cars of a four-cars VIRM, and their induced vibration, are illustrated in Figure 4-6.

In addition to types of trains and their induced responses, other data such as number of passing axles per month, total tonnage per month, and total number of passing trains per month were also available.

4-2 Experiments

Considering the dataset from the case study of a crossing explained in Section 4-1, following experiments were performed:

1. Extracting input representative features (types of trains classification).
   (a) Types of trains classification using fully shared convolutional networks and transient responses from different accelerometers:
      i. Classification with transient responses from Accelerometer 1 (accelerometer with the highest magnitude).
      ii. Classification with transient responses from Accelerometer 1, 2, and 4 (all accelerometers located bellow the crossing nose).
   (b) Types of trains classification using various Convolutional Neural Networks (CNN) configurations:
      i. Classification with separated convolutional networks.
      ii. Classification with two parallel convolutional networks, where the first convolutional network was used for transient responses from Accelerometer 1, and the other convolutional network was shared between transient responses from Accelerometer 2 and 4.
   (c) Types of trains classification with other states of the art classifiers (Support Vector Machine (SVM) and K-Nearest Neighbors (K-NN)):
      i. Using statistical values derived from transient responses as the input.
      ii. Using the wavelet time-frequency entropy extracted from the wavelet coefficients as the input.
2. Extracting defect representative features (localization maps).

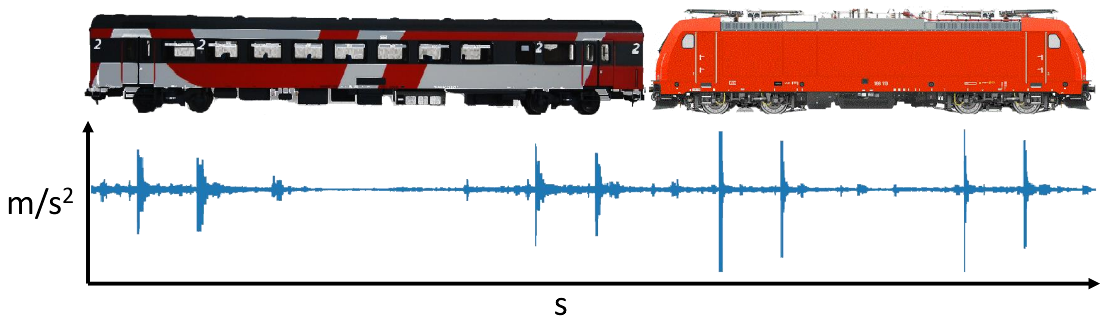


Figure 4-4: The vibration signal relates to the vehicle structure of a TRAXX locomotive and an ICRm coach

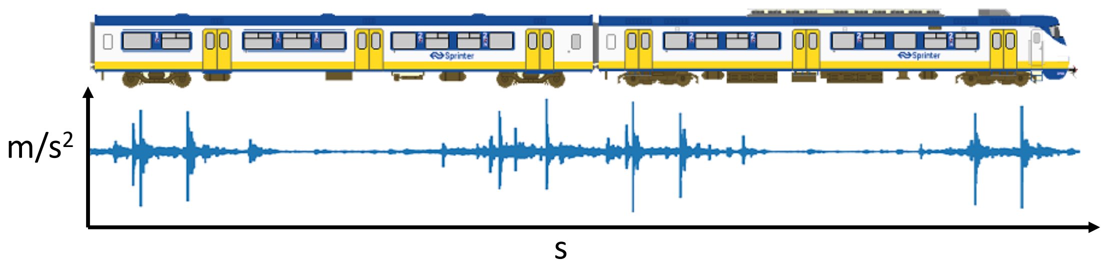


Figure 4-5: The vibration signal relates to the vehicle structure of an SGMm train (SGM-III).

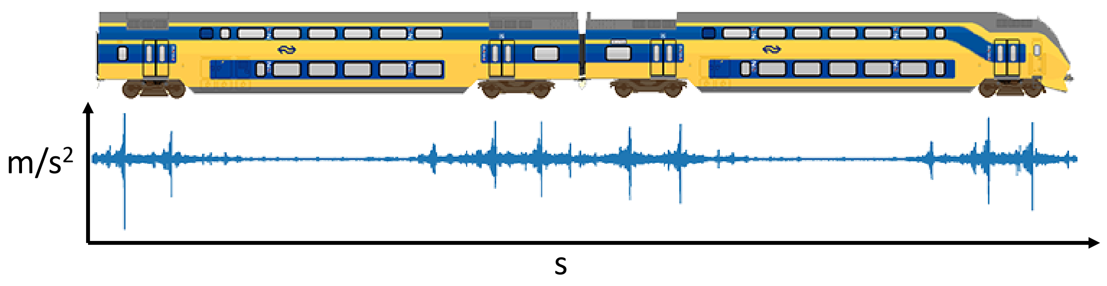


Figure 4-6: The vibration signal relates to the vehicle structure of a four-cars VIRM train.

   (a) Regression test based on the number of passing axles using different length of transient responses:
      i. Transient responses with the period of 0.625 ms (only the peak).
      ii. Transient responses with the period of 10 ms (full transient response).
   (b) Localization maps generation using transient responses with the best performing period and various areas of interest.
      i. The size of last average-pooling layer was 1×8, focusing on the time domain.
      ii. The size of last average-pooling layer was 3×4, focusing on both time and frequency domain.
      iii. The size of last average-pooling layer was 6×1, focusing on the frequency domain.

3. Estimating degradation level of the crossing.
   (a) Regression test with and without compensating features:
      i. Regression test with defect representative features (without input representative features) and fully convolutional networks (ConvNet).
      ii. Regression test with all features (input and defects representative features) and fully convolutional networks (ConvNet).
   (b) Regression test with different CNN architectures:
      i. Regression test with all features and inception networks consisting of convolutional and average-pooling layers arranged in parallel.
      ii. Regression test with all features and inception networks consisting of convolutional and max-pooling layers arranged in parallel.
      iii. Regression test with all features and inception networks consisting of convolutional, average-pooling layers, and max-pooling layers arranged in parallel.
   (c) Regression test with different numbers of convolutional depths and filters:
      i. Regression test with all features and the best performing architecture where the depth of convolutional networks vary.
      ii. Regression test with all features and the best performing architecture where the number of convolutional filters vary.

All experiments were carried out using a personal computer with the specification as follows:

- Processor: Intel Core i7-4710HQ CPU @ 2.5 GHz (8 CPU)
- Memory: 16 GB DDR3 PC-12800 (800 MHz)
- Graphics: NVIDIA Geforce GTX 860M (4 GB)

4-3 Results

4-3-1 Input representative features

The input representative features were extracted using CNN and other states of the art classifiers such as SVM and K-NN so that their performances could be compared.

Types of trains classification using fully shared convolutional networks and transient responses from different accelerometers

Two fully shared CNN classifiers were utilized for classifying types of train. The one fed with wavelet coefficients calculated from signals produced by Accelerometer 1 (**FSh1-CNN**), and the other one fed with wavelet coefficients calculated from signals produced by Accelerometer 1, 2, and 4 (**FSh2-CNN**). As seen from the confusion matrices presented in Figure 4-8 (a) and (b), the overall accuracy achieved by FSh1-CNN was 62.09%, and the one achieved by FSh2-CNN was 74.16%. FSh2-CNN also outperformed FSh1-CNN in all performance indices as shown in Figure 4-9. FSh1-CNN produced prediction results with lower accuracy because the transient responses were also affected by dynamical changes of the crossing over time. The indication can be seen from Figure 4-7 (a) to (c), which shows that the majority of incorrect predictions were contributed by wavelet coefficients above 1500 Hz. Transient responses from the higher frequency domain are prone to the changes due to the existence of defects [47]; this will be discussed in Section 4-3-2. In the case of FSh2-CNN, the classifier could achieve a higher accuracy because it utilized more information gathered by Accelerometer 2 and 4, which were placed further from the location of the wheel-crossing impact compared to Accelerometer 1. Since Accelerometer 2 and 4 produced responses with lower magnitude and frequency compared to Accelerometer 1, they were less affected by changes of responses in the higher frequency domain.

Types of trains classification using various CNN configurations

In this experiment, different CNN configurations were utilized: fully separated convolutional networks **(FS-CNN)** and two parallel (semi-shared) convolutional networks **(2S-CNN)** where the first convolutional network was used by transient responses from Accelerometer 1, and the other convolutional network was used by transient responses from Accelerometer 2 and 4. Details of these architectures are presented in Appendix B. Confusion matrices of these experiments are shown in Figure 4-8 (b) and (c). Overall, 2S-CNN achieved the highest accuracy (93.40%) among other classifiers fed with wavelet coefficients calculated from transient responses produced by Accelerometer 1, 2, and 4, followed by FS-CNN (81.78%), and FSh2-CNN (74.16%). From the performance measures shown in Figure 4-9, TRAXX had the highest score of accuracy and specificity, which means that this class mostly contributed to the accuracy of predictions. Moreover, least numbers of false alarm were triggered by TRAXX compared to the other trains. VIRM was the type of trains that is most recognizable by the classifiers, as shown by high numbers of precision in many experiments, followed by ICRM and TRAXX. SGMm had the lowest values of recall and precision because according to most classifiers this type of trains is similar to ICRm as shown in confusion matrices presented in Figure 4-8.

Types of trains classification with other states of the art classifiers

Two kinds of inputs were fed into classifiers to perform types of trains classification: statistical values calculated from transient responses and the wavelet entropy calculated from wavelet coefficients. Confusion matrices obtained from classifications performed by linear SVM, quadratic SVM, and K-NN were presented in Figure 4-10. Overall, quadratic SVM fed with wavelet entropy (**Q-SVM W**) produced the best accuracy (59.54%), followed by quadratic SVM fed with statistical values (**Q-SVM S**) (58.45%), and K-NN fed with wavelet entropy (**K-NN W**) (53.47%).

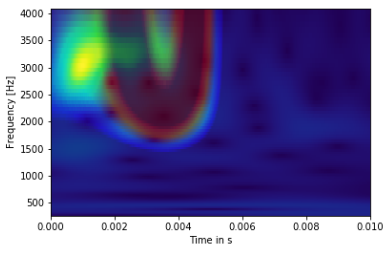
Other benchmarking parameters are shown in Figure 4-11. In most classifiers, TRAXX contributed the most of true prediction shown by high numbers of recall, followed by VIRM. TRAXX also contributed to the least number of false alarms as can be seen from the specificity bar chart. In contrast, VIRM had the lowest values of specificity. TRAXX also had the highest rate of true prediction compared to other types of train seen from the recall bar chart.

It is seen from both confusion matrices shown in Figure 4-10 and performance measures shown in Figure 4-11 that machine learning classifiers such as SVM and K-NN are not suitable for classifying types of trains from a transient response induced by a wheel-crossing impact. Transient responses are not linearly separable, compared to other physical features such as distances between wheels representing the speed of trains and the length of coaches [44]. In order to prove this argument, distances between wheels extracted from the case study were fed to the machine learning classifiers. The overall accuracy of 87.4% could be achieved by Q-SVM, followed by K-NN (83.07%), and L-SVM (78.69%).

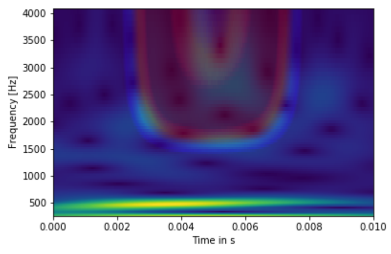
Comparison of the overall accuracy achieved by all classifiers is shown in Figure 4-12, which shows that all CNN classifiers outperformed both SVM and K-NN, and the highest accuracy was achieved by 2S-CNN. Additionally, performance measures of CNN, SVM, and K-NN are listed on Table 4-1.

4-3-2 Defects representative features (localization maps)

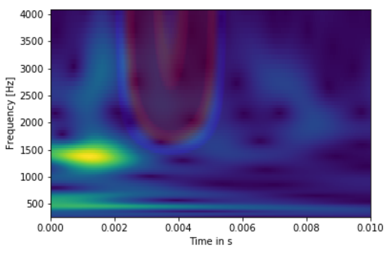
Extracting defect representative features was started by performing a regression test to follow an increasing function according to a transient response induced by the wheel-crossing impact. The purpose is to build the localization map showing the area of wavelet coefficient matrices that contribute to the input. Based on the study in [13] mentioned in



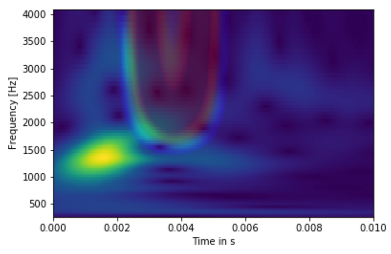
(a)



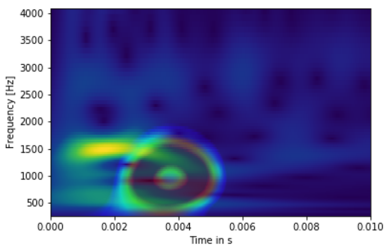
(b)



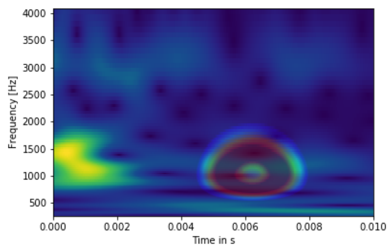
(c)



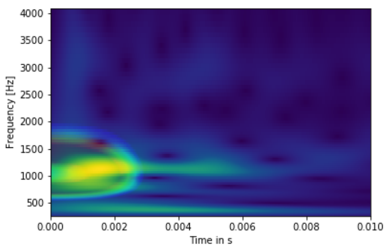
(d)



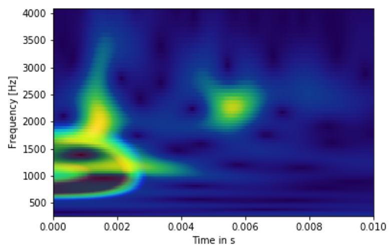
(e)



(f)



(g)



(h)

Figure 4-7: Areas of wavelet coefficients covered by the reddish shapes are the areas that have the most contribution to the results of predictions: (a) a VIRM was detected as an SGMm; (b) an ICRm was detected as a TRAXX; (c) an SGMm was detected as an ICRm; (d) the correct prediction of an ICRm; (e) the correct prediction of an ICRm; (f) a correct prediction of an SGMm; (g) the correct prediction of a TRAX; (h) the correct prediction of a VIRM. As seen from (a) to (c), the most incorrect predictions came from features in the high frequency domain.

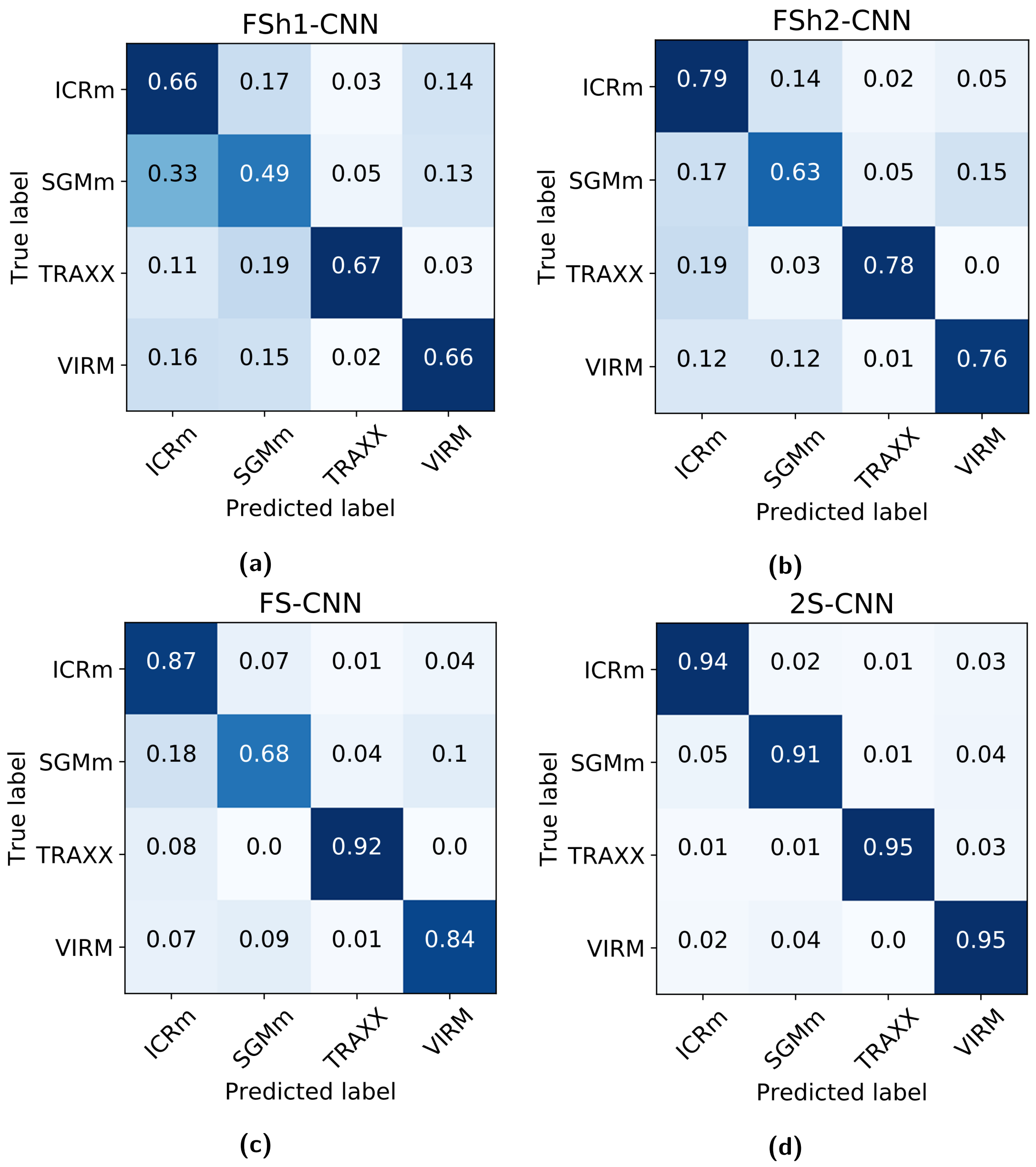
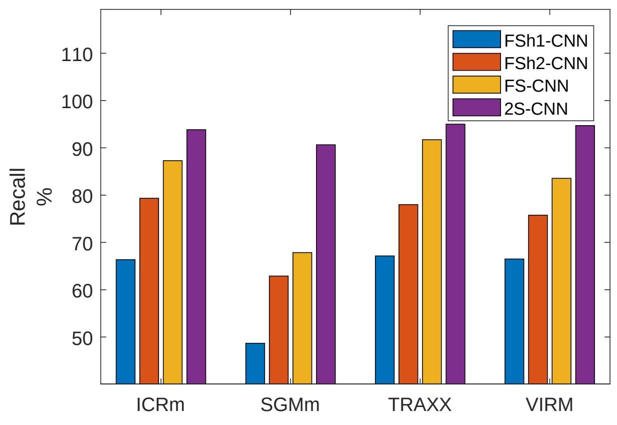
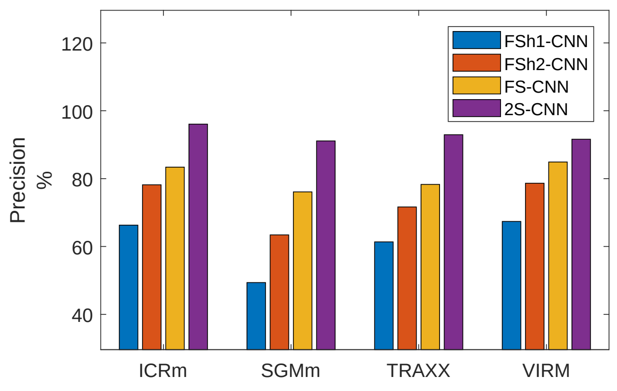


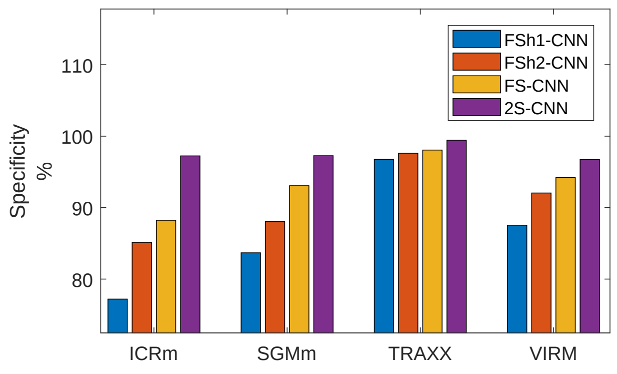
Figure 4-8: Results of classifying types of trains using 4 CNN configurations: (a) fully shared convolutional networks with transient responses from Accelerometer 1; (b) fully shared convolutional networks with transient responses from Accelerometer 1, 2, and 4; (c) fully separated convolutional networks with transient responses from Accelerometer 1, 2, and 4; (d) two separated convolutional networks. The first convolutional networks used by transient responses from Accelerometer 1, and the second convolutional network shared by transient responses from Accelerometer 2 and 4.



(a)



(b)



(c)

Figure 4-9: Performance measures of CNN classifiers: (a) recall, (b) precision, and (c) specificity.

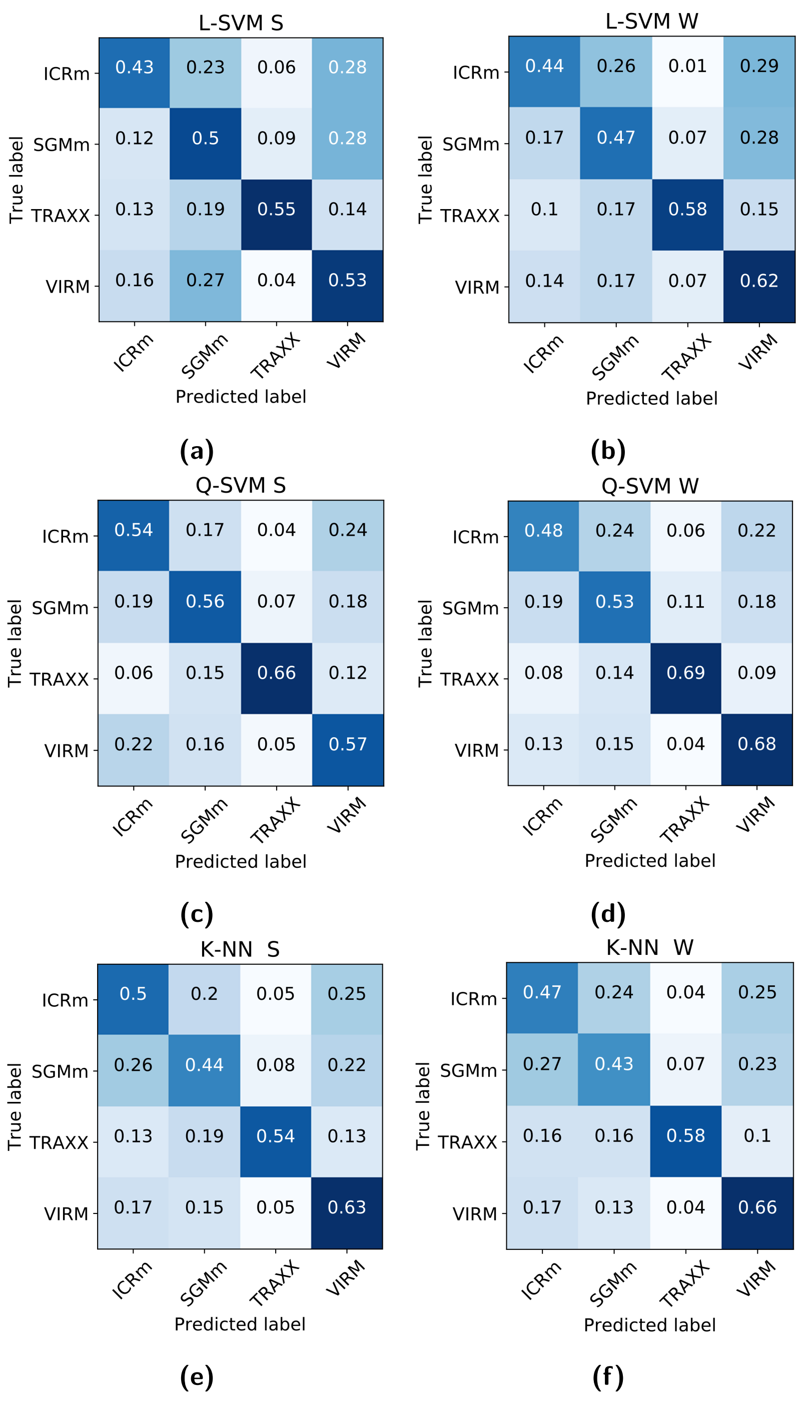
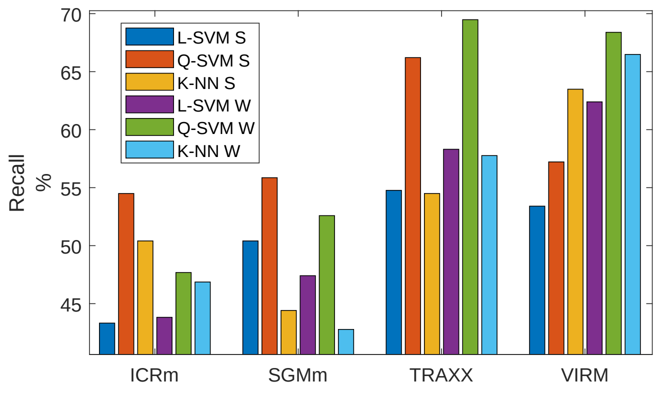
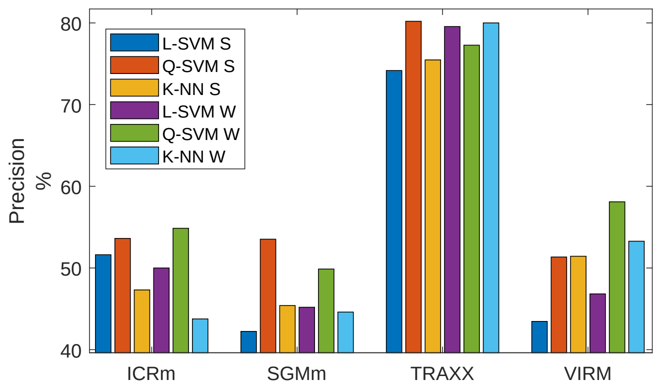


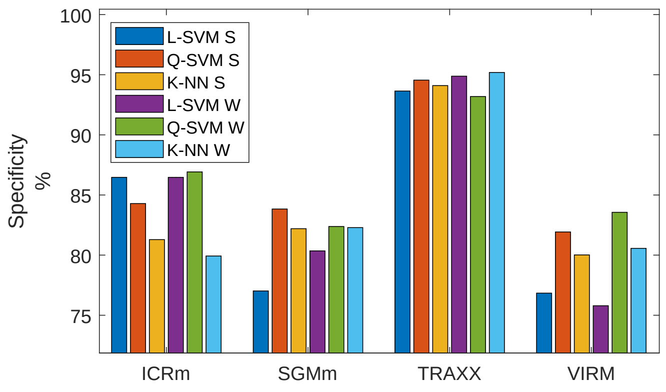
Figure 4-10: Results of classifications with other state of the art methods: (a) linear SVM fed with statistical values (L-SVM S), (b) linear SVM fed with wavelet time entropy (L-SVM W), (c) quadratic SVM fed with statistical values (Q-SVM S), (d) quadratic SVM fed with wavelet time entropy (Q-SVM W), (e) K-NN fed with statistical values, and (f) K-NN fed with wavelet time entropy.



(a)



(b)



(c)

Figure 4-11: Performance measures for the baseline classifiers: (a) recall, (b) precision, and (c) specificity.

Table 4-1: Performance indices of all classifiers.

Classifier	Type of trains	Precision (%)	Recall (%)	Specificity (%)	Accuracy (%)
L-SVM S	ICRm	51.62	**43.32**	86.47	**50.48**
	SGMm	**42.24**	50.41	77.02	
	TRAXX	**74.17**	**54.77**	**93.64**	
	VIRM	43.46	53.41	**76.84**	
Q-SVM S	ICRm	53.62	**54.50**	84.29	58.45
	SGMm	53.52	55.86	83.83	
	TRAXX	**80.20**	**66.21**	**94.55**	
	VIRM	**51.34**	57.22	**81.93**	
K-NN S	ICRm	47.31	50.41	81.29	53.20
	SGMm	**45.40**	**44.41**	82.20	
	TRAXX	**75.47**	54.50	**94.10**	
	VIRM	51.43	**63.49**	**80.02**	
L-SVM W	ICRm	50.00	**43.82**	86.47	53.16
	SGMm	**45.19**	47.41	80.35	
	TRAXX	**79.55**	58.31	**94.88**	
	VIRM	46.83	**62.40**	**75.79**	
Q-SVM W	ICRm	54.86	**47.68**	86.92	59.54
	SGMm	**49.87**	52.59	82.38	
	TRAXX	**77.27**	**69.48**	**93.19**	
	VIRM	58.10	68.39	**83.56**	
K-NN W	ICRm	**43.77**	46.87	**79.93**	53.47
	SGMm	44.60	**42.78**	82.29	
	TRAXX	**80.00**	57.77	**95.19**	
	VIRM	53.28	**66.49**	80.56	
FSh1-CNN	ICRm	66.30	66.36	**77.20**	62.09
	SGMm	**49.37**	**48.68**	83.68	
	TRAXX	61.36	**67.14**	**96.75**	
	VIRM	**67.38**	66.50	87.54	
FSh2-CNN	ICRm	78.18	**79.34**	**85.14**	74.16
	SGMm	**63.42**	**62.89**	88.04	
	TRAXX	71.65	78.00	**97.62**	
	VIRM	**78.64**	75.75	92.05	
FS-CNN	ICRm	83.39	87.29	88.23	81.78
	SGMm	**76.09**	**67.85**	**93.06**	
	TRAXX	78.29	**91.71**	**98.06**	
	VIRM	**84.90**	83.55	94.23	
2S-CNN	ICRm	**96.06**	93.84	97.24	**93.4**
	SGMm	**91.11**	**90.65**	97.26	
	TRAXX	92.93	**95.00**	**99.44**	
	VIRM	91.61	94.71	**96.73**	

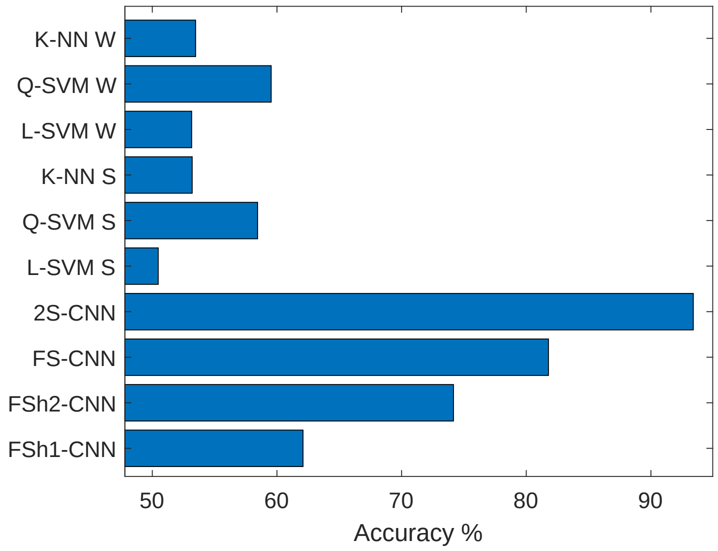


Figure 4-12: Overall accuracy of CNN and other states of the art classifiers.

Section 2-2, the number of passing axles can be used to predict when the cracks initiate on the surface of crossing nose. Since the number of passing axles per month is available in the dataset, an increasing function, where the input is wavelet coefficients and the output is the total number of axles that have passed the crossing, was created from the data for the regression test.

Regression test based on the number of passing axles using different length of transient responses

Wavelet coefficients with two different lengths were utilized for the regression test so that the length of signals used in further processing could be decided. Comparing both results, the input utilizing full transient responses (t = 10 ms) have the better performance in tracking the normalized number of passing axles compared to the input using only the peaks (t = 0.625 ms) as shown in Figure 4-13. High deviation in the signal with only the peak could be caused by deteriorated contact surfaces of the train wheels or due to the effect of faster trains as explained in [64]. With full transient responses, information such as the time constant is preserved in wavelet coefficients. In the further processing, wavelet coefficients from full transient responses are utilized.

Localization map generation using transient responses with the best performing period and various area of interests

Localization maps with three different areas of interests as shown in Figure 3-12 were constructed using transient responses with the length of 10 ms to show their evolution.

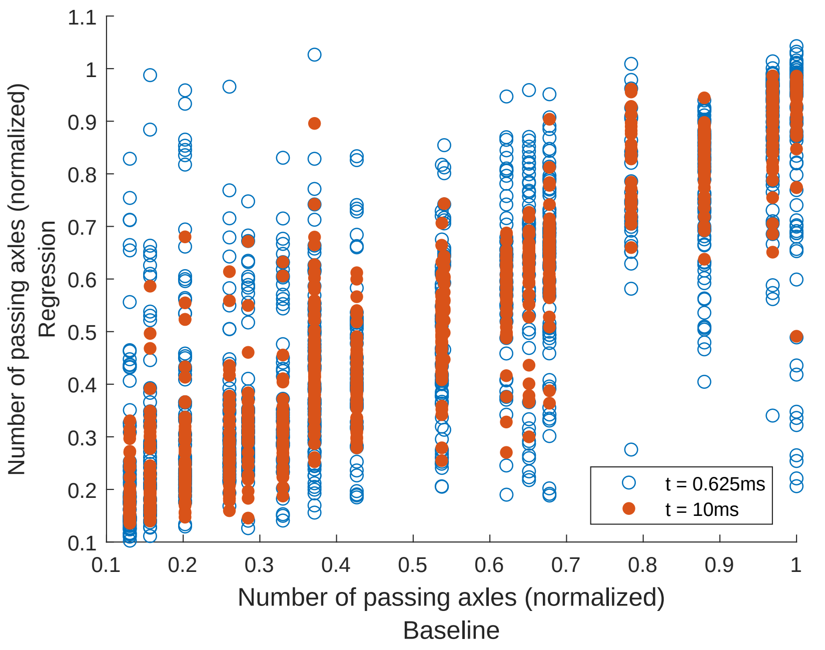
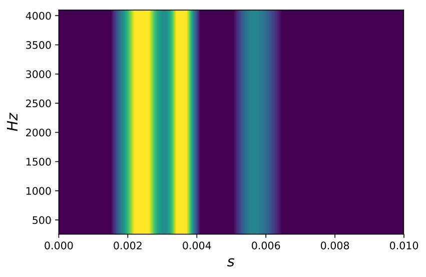


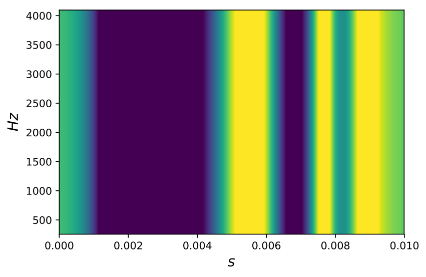
Figure 4-13: Results of regression test to fit the number of passing axles (normalized) based on the wavelet coefficients. Signals with full responses produced better results compared to signals with only the peaks.

As shown in Figure 4-14, random locations from the time domain are highlighted by the localization maps. As there is no constant highlight in a particular area of the wavelet coefficients in the time domain, there was no apparent indication of features regarding the defects that could be seen from the time domain. In Figure 4-15, similar to what is shown in Figure 4-14, there is no apparent indication of features that indicate the defects in the time domain. However, it could be seen that the region at high-frequency responses started to contribute mostly from June 2015 onward. This indication is shown by the constant highlight of the wavelet coefficients in the area above 1.5 kHz. The most obvious indication of the defects can be seen from Figure 4-16. Using the configuration shown in Figure 3-11 (c), the responses between $2.5 - 3$ kHz were constantly highlighted from June 2015 onward. Since small surface defects induce high-frequency responses, and similar region of the wavelet coefficients in Figure 4-15 were also highlighted from June 2015 onward, we can suspect that this frequency band indicates the defects.

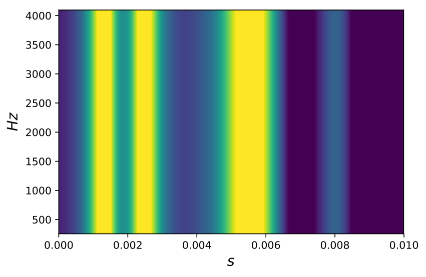
In order to confirm the results, similar indications should be seen directly from the wavelet coefficients as well. As parts of areas highlighted by the localization maps in the particular frequency domain (all areas in the time domain were highlighted), maximum values of the wavelet coefficients within $2.5 - 3$ kHz, including all region at the time domain ($0 - 10$ ms), were obtained and averaged per measurement. As shown in Figure 4-17, the wavelet coefficient between $2.5 - 3$ kHz also indicated a positive trend. This



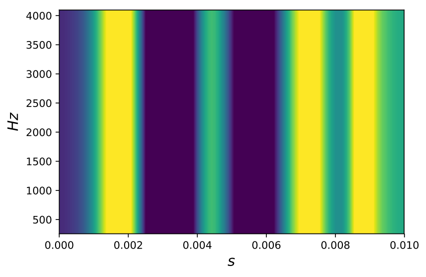
(a) 12/03/2015



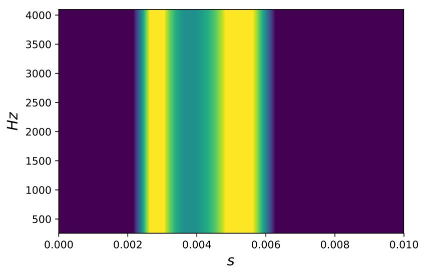
(b) 16/04/2015



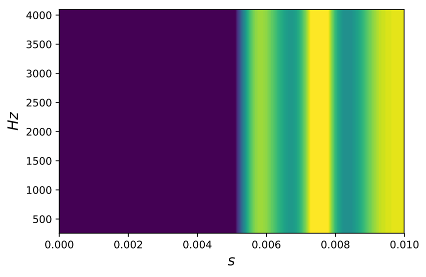
(c) 11/05/2015



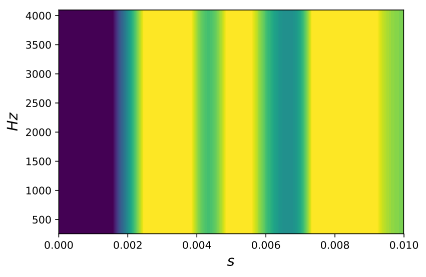
(c) 29/06/2015



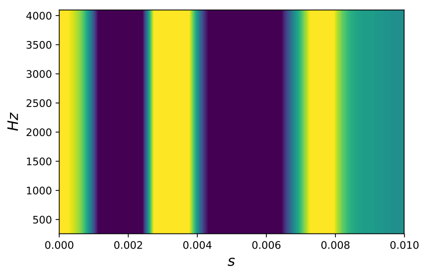
(d) 22/07/2015



(e) 6/08/2015

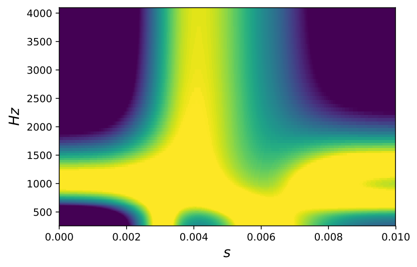


(f) 29/09/2015

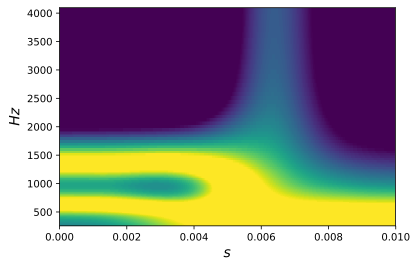


(g) 30/10/2015

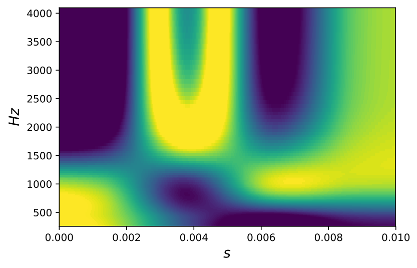
Figure 4-14: Evolution of the transient responses according to the localization maps focusing on the time domain (1×8). The time domain area cannot show the indication of defects as there is no constant highlight in a particular area of the wavelet coefficients when defects existed.



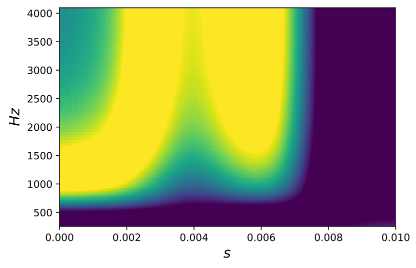
(a) 12/03/2015



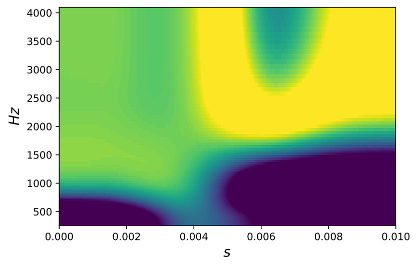
(b) 16/04/2015



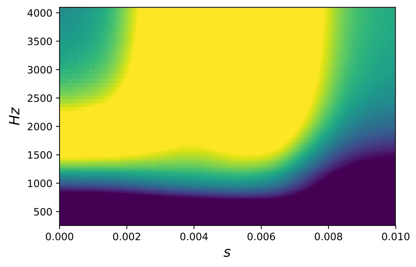
(c) 11/05/2015



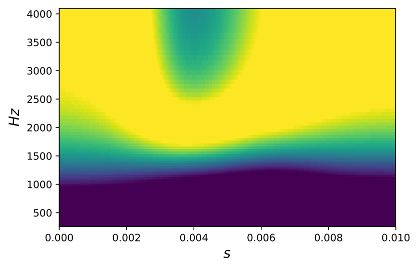
(d) 29/06/2015



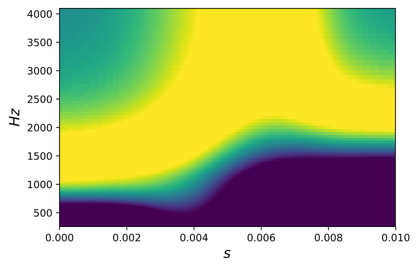
(e) 22/07/2015



(f) 6/08/2015

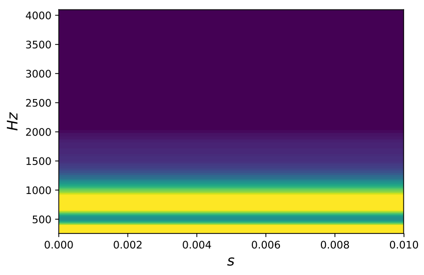


(g) 29/09/2015

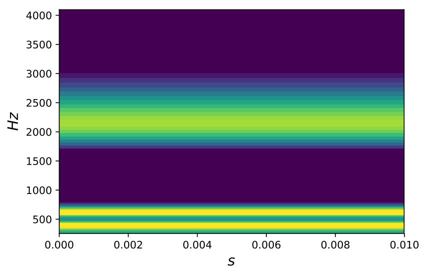


(h) 30/10/2015

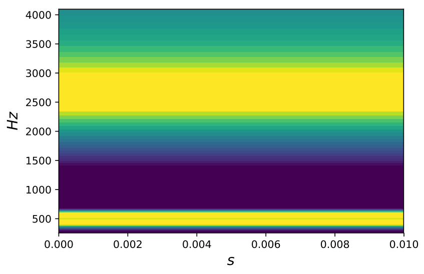
Figure 4-15: Evolution of the transient responses according to the localization maps focusing on both time and frequency domain (3×4). The area on the time domain was not constantly highlighted. However, the area above 1.5 kHz was constantly highlighted since June 2015. There is an indication of defects located in the higher frequency domain.



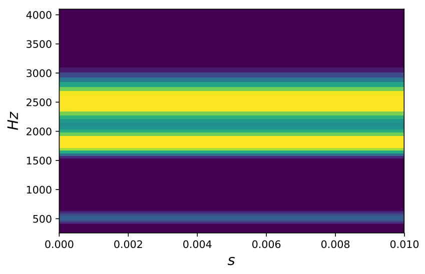
(a) 12/03/2015



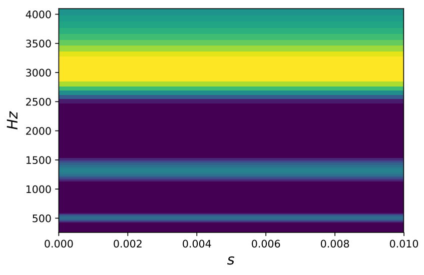
(b) 16/04/2015



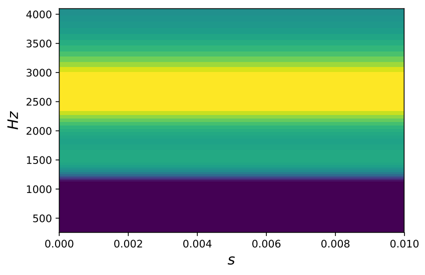
(c) 11/05/2015



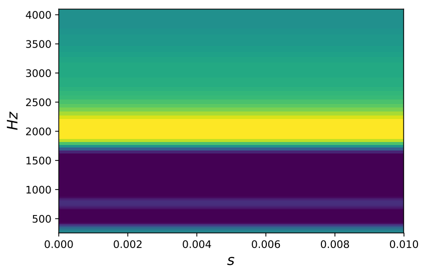
(d) 29/06/2015



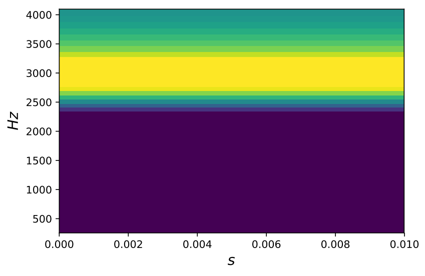
(e) 22/07/2015



(f) 6/08/2015



(g) 29/09/2015



(h) 30/10/2015

Figure 4-16: Evolution of the transient responses according to the localization maps focusing on the frequency domain (6×1). It is clear that the region around 2.5-3 kHz was constantly highlighted from May 2015, which can be the features that indicate the defects

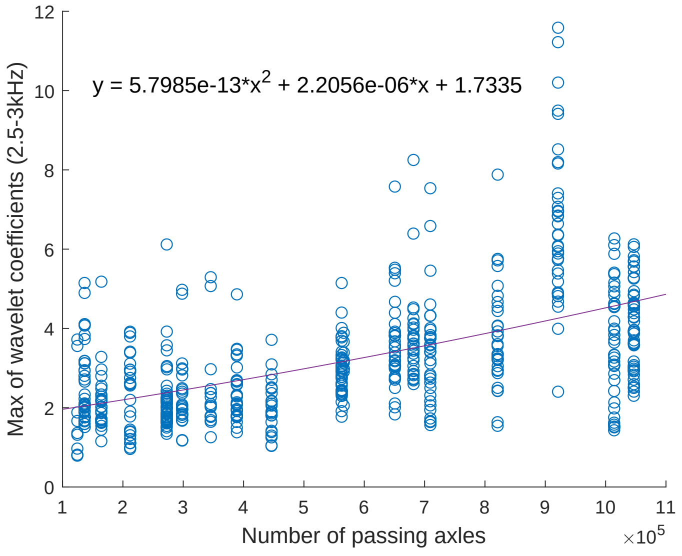


Figure 4-17: Average of the maximum wavelet coefficient $(2.5 - 3kHz)$ per measurement.

means that we could use wavelet coefficients in the range of $2.5 - 3$ kHz as the defect representative features.

4-3-3 Degradation level of the crossing

The degradation level of the crossing is estimated using a CNN regressor. In order to find the best performing architecture and configuration suitable for this task, several CNN architectures are tested. Then, the best performing architecture is picked and tested again with different configurations such as different depth of convolutional layer and number of convolutional features. The output of all architecture is also compared to find out which architecture provided the best estimation.

Regression test with and without compensating features

The first experiment was to find out whether the input representative features could give a positive effect to the CNN regressor or not. In order to perform this task, the trendline obtained from the previous step shown in Figure 4-17 was utilized to construct baseline values representing the actual degradation level of the crossing based on the number of passing axles. As shown in Figure 4-18 (a), the degradation level of regression had a smaller deviation compared to the degradation level obtained from the wavelet coefficients. Moreover, a CNN regressor utilizing input representative features outperformed the one without input representative features even if they are unnoticeable, as shown in Figure 4-18 (b). The comparison of Mean Squared Error (MSE) between the ConvNet implementing the input representative features and the ConvNet without the input representative features were 0.026 and 0.044 respectively. Even the gap between two MSE is small, the ConvNet implementing the input representative features performs better than the ConvNet without the input representative features.

Regression test with different CNN architectures

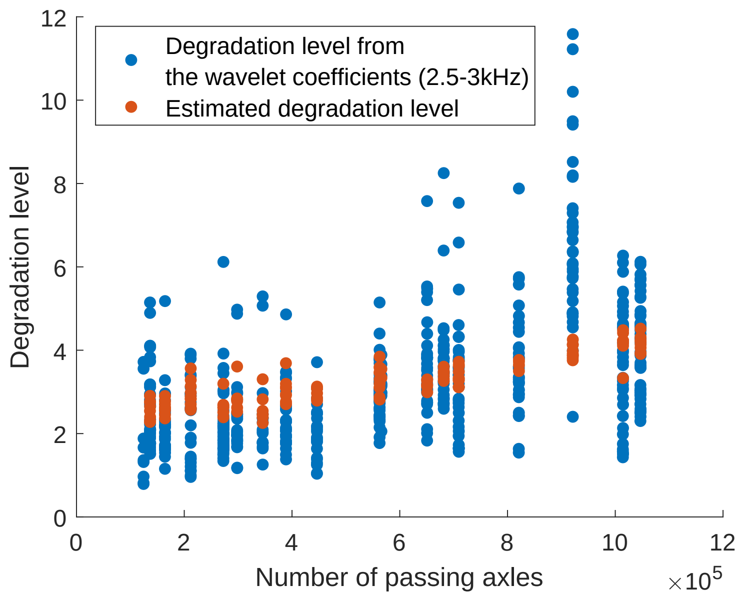
Four CNN architectures such as full convolutional networks (**ConvNet**), convolutional networks with average-pooling inception networks (**Inception Conv + Avg**), convolutional networks with max-pooling inception networks (**Inception Conv + Max**), and convolutional networks with average and max-pooling inception networks (**Inception Conv + Avg+ Max**) as shown in Figure 3-14 to 3-17, were utilized to estimate the degradation level of the crossing. Details of these architectures are presented in Table B-4 and Table B-5 of Appendix B, and the benchmarking parameters are presented in Figure 4-19 to 4-20. In the case of MSE values, ConvNet outperformed Inception networks. ConvNet also contains simpler mathematical operations compared to all Inception networks containing at least one max or average-pooling layer, which is expensive, making ConvNet as the fastest architecture to perform prediction. However, ConvNet took a longer time to train because it contains more convolutional filters performing sliding windows operation compared to all Inception networks, which share the number of filters with the average/max-pooling layer.

Comparing the performance of three Inceptions, Inception Conv + Max produced the least MSE values, followed with Inception Conv + Avg, and Inception Conv + Avg + Max. In the case of learning time, Inception Conv + Avg + Max took the longest time to be trained because it had the highest number of trainable parameters, followed by Inception Conv + Avg, and Inception Conv + Max. Even Inception Conv + Avg and Inception Conv + Max had the same number of trainable parameters, Inception Conv + Avg took a longer time to be trained because average-pooling is more expensive operation than max-pooling.

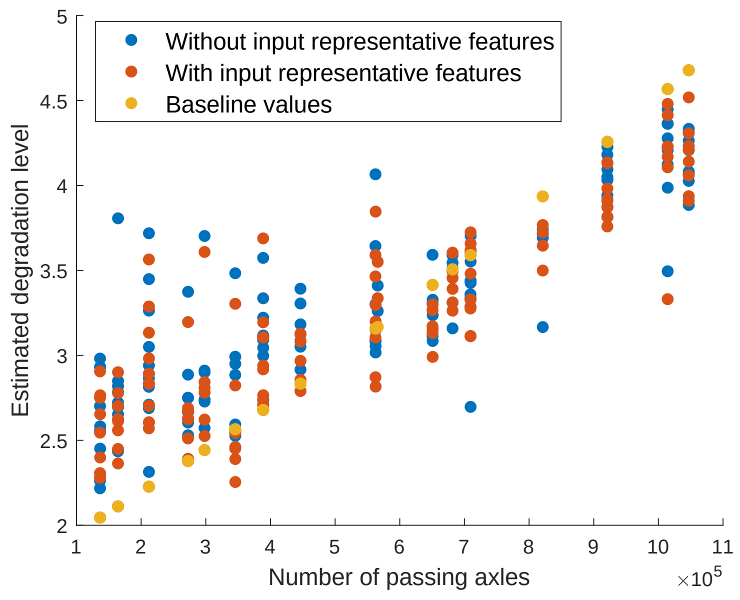
Regression test with different numbers of convolutional depths and filters

In order to obtain the best configuration, various depths of convolutional layers were utilized to run the regression tests. The depths of convolutional layers were defined as follows: ConvNet consisted of 5 convolutional networks, ConvNet D4 consisted of 4 convolutional networks, and ConvNet D3 consisted of 3 convolutional networks.

Details of these configurations can be seen in Table B-2 of Appendix B. As seen from the results shown in Figure 4-19 to 4-20, ConvNet outperformed ConvNet D4 and ConvNet D3 in terms of the MSE values. This result matches with other experiments showing that the number of convolutional layers has a great affection to the accuracy of the prediction [59]. ConvNet also learned 175.9 s faster compared to ConvNet D3 because it has fewer trainable parameters. ConvNet D4 was slightly faster 35.6 s than ConvNet since it also



(a)



(b)

Figure 4-18: Degradation level of the crossing: **(a)** comparison of the degradation level obtained from wavelet coefficients and the one estimated by the CNN algorithm; **(b)** comparison of the estimated degradation level obtained with and without input representative features.

Table 4-2: Benchmarking parameters of all CNN architectures

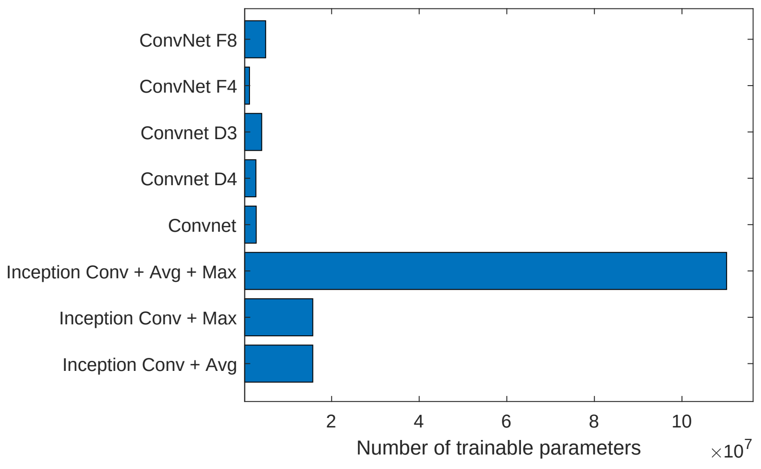
Architectures	**Learning time** (s)	**Prediction time** (s)	**Number of trainable parameters**	**MSE**
Inception Conv + Avg	2708.33	**87.93**	15696093	0.0327
Inception Conv + Max	**2390.04**	80.22	15696093	0.0305
Inception Conv + Avg + Max	**4151.17**	80.05	**110198807**	0.0349
ConvNet	3596.73	79.22	2781187	**0.0203**
ConvNet D4	3561.12	78.44	2707171	0.0246
ConvNet D3	3737.03	76.45	4034207	0.0253
ConvNet F4	3473.89	**73.46**	**1241129**	**0.0406**
ConvNet F8	3648.4	81.32	4934269	0.0379

has fewer trainable parameters. Regarding the prediction time, ConvNet was the slowest architecture with the difference of 1 s with ConvNet D4, and 2 s with ConvNet D3. The sliding windows mechanism used by convolutional layers to produce filter maps is slower than vector multiplications performed in a fully connected multi-layer perceptron. Since ConvNet had the deepest depth of convolutional layers, it processed the inputs slower than ConvNet D4 and ConvNet D3.

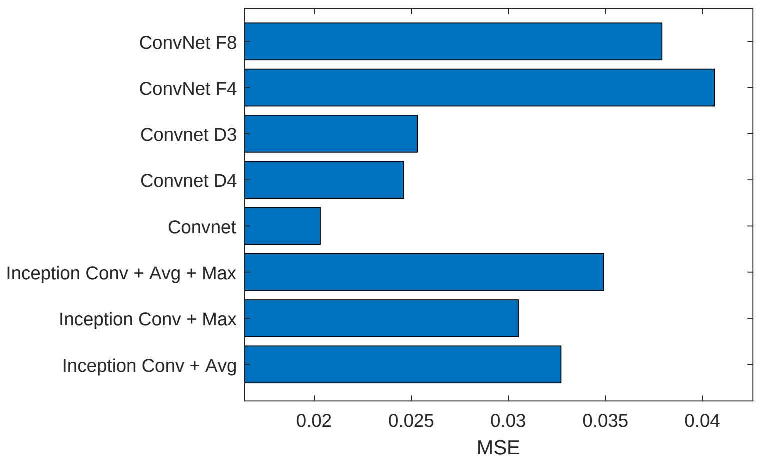
Various numbers of filters were also experimented to find the most suitable algorithm as follows: ConvNet consisted of 6 filters in the first convolutional network, and multiplied by 2 in each next convolutional network (same architecture as before); ConvNet F4 consisted of 4 filters in the first convolutional network, and multiplied by 2 in each next convolutional network; ConvNet F8 consisted of 8 filters in the first convolutional network, and multiplied by 2 in each next convolutional network.

Details of these configurations can be seen in Table B-3 of Appendix B. The fastest architecture for both learning and predicting was ConvNet F4 as it had the fewest number of filters. However, ConvNet F4 also had the worst MSE. Results of this experiment match with another experiment showing that as the number of convolutional filters increases, the accuracy also increases (i.e. MSE decreases) [59]. However, this relationship also depends on the number of outputs because an excessive number of convolutional filters could cause the number of trainable parameters explode. Since in this experiment CNN algorithms were used for regression tests producing a single output, the number of parameters should be kept fitted. In this case, ConvNet had the most suitable number of convolutional filters so that it produced the lowest MSE compared to the architecture with the smallest number of convolutional filters (ConvNet F4) and the one with the highest number of filters (ConvNet F8).

Using CNN architectures from ConvNet and all Inception networks, results of health estimations are presented in Figure 4-21. Surface defects were visually spotted in August 2015, after around 700000 axles had passed the crossing (red box). Using ConvNet and all Inception networks, the symptoms were detected could have been seen between May and June 2015 (the estimated degradation level was about 2.7) after about 450000 passing axles (yellow box), giving about two months extra to the maintenance engineers to plan

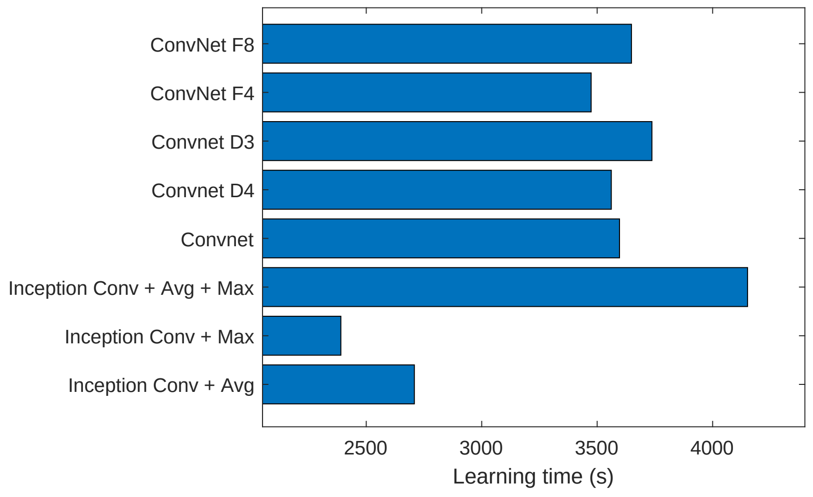


(a)

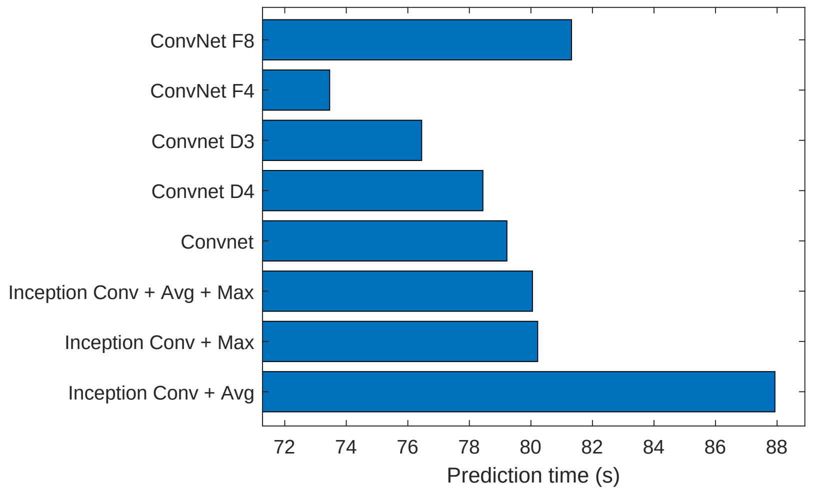


(b)

Figure 4-19: Benchmarking parameters for all CNN architectures: (a) number of trainable parameters, (b) MSE

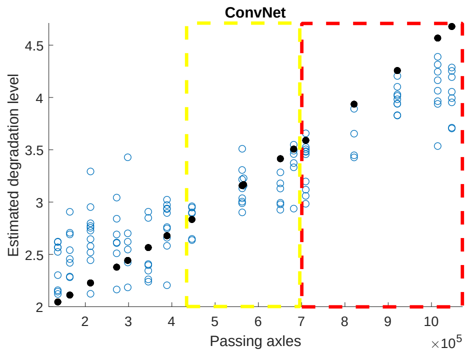


(a)

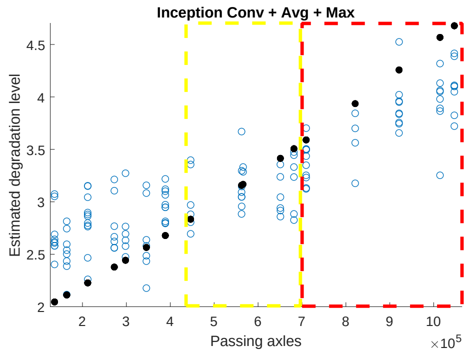


(b)

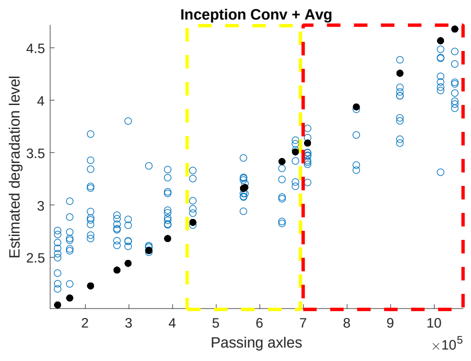
Figure 4-20: Benchmarking parameters for all CNN architectures: (a) learning time, (b) prediction time



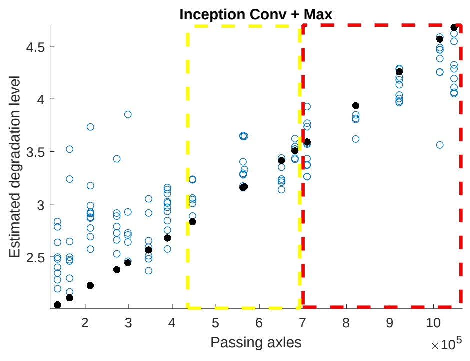
(a)



(b)



(c)



(d)

Figure 4-21: Results of estimated degradation level of the crossing produced by all CNN architectures. The degradation level of 2.7 was considered as the threshold to determine defects in the crossing, which was chosen because there was no estimation below this value after about 450000 passing axles.

maintenance activities. The degradation level of 2.7 was chosen because there was no estimation below this threshold after about 450000 passing axles.

Regarding all architectures, ConvNet is the most suitable CNN architecture for estimating the degradation level of crossing, followed by Inception Conv + Max, Inception Conv + Avg + Max, and Inception Conv + Avg. ConvNet produced the least number of MSE. ConvNet did not produce many estimations above the baseline values (black circles) when the number of passing axles was below 450000 compared to Inception Conv + Avg and Inception Conv + Max. An estimation that is too optimistic can trigger a false alarm. Both ConvNet and Inception Conv + Max did not produce any estimation below the baseline values as many as Inception Conv + Avg and Inception Conv + Avg+ Max, when the number of passing axles was above 450000. Many estimations above the baseline values may trigger many false alarms. In contrast, many estimations below the baseline values may cause many miss detections. Therefore, choosing CNN architecture having the least MSE is crucial. The conclusion is that ConvNet is the most suitable

CNN architecture for estimating the degradation level of crossing.

4-4 Summary

This chapter discussed the evaluation of the methodology for monitoring the health condition of railway crossings: the dataset from the case study of a crossing, the evaluation, and results.

In order to evaluate the proposed methodology, the dataset from the case study of a crossing located in Amsterdam was utilized. The dataset contains the responses of wheel-crossing impacts recorded by several accelerometers, which are installed on the crossing, with the sampling frequency of 102.4 kHz. The acceleration data is also labeled with several types of trains, four of them are used in this thesis: TRAXX, ICRm, SGMm, and VIRM. Moreover, additional data such as the number of passing axles per month is also available along with the dataset.

Three main experiments were performed in order to evaluate the proposed methodology: types of trains classification, localization maps, and regression tests for estimating the health condition of the crossing. In types of trains classification, various CNN configurations were used: fully-shared CNN with only one accelerometer input (FSh1-CNN), fully-shared CNN with three accelerometer input (FSh2-CNN), fully-separated CNN with three accelerometer input (FS-CNN), and semi-separated CNN with 2 convolutional networks (one network was dedicated for one accelerometer, and another network was shared for two accelerometers) (2S-CNN). Additionally, three machine learning classifiers were also utilized for comparison: linear SVM (L-SVM), quadratic SVM (Q-SVM), and K-NN. Two different features are used for the machine learning classifiers: statistical values derived directly from the transient responses and the wavelet time entropy. Overall, all CNN classifiers outperformed the machine learning classifiers. The best performing CNN classifier was 2S-CNN with 93.4% of accuracy. Among the machine learning classifiers, Q-SVM, when it is fed with the wavelet time entropy, is the best classifier with 59.54% accuracy.

In the localization maps, three areas of interest are defined to find out the areas of wavelet coefficients that can be used as features that indicate defects. The first areas of interest focus on the time domain, the second areas of interest focus on both the time domain and the frequency domain, and the third areas of interest focus on the frequency domain. In order to generate the localization maps, information regarding the total number of passing axles per month was used as the trend for the regression tests. From all chosen areas of interests, it can be concluded that the features that indicate defects could not be detected from the time domain. The features that indicate defects were detected in the frequency domain between 2.5-3 kHz. When this range of frequency is inspected directly from the wavelet coefficients, a degradation model with many deviations could be built.

In estimating the degradation level of the crossing, three different CNN architectures were utilized: the fully connected convolutional networks (ConvNet), the inception networks with average-pooling networks (Inception Conv + Avg), inception networks with max-pooling networks (Inception Conv + Max), inception networks with average-pooling and max-pooling networks (Inception Conv + Avg + Max). Moreover, the number of filters and convolutional depths are also set to different values in order to find the best estimator. Overall, The ConvNet is the best architecture for estimating the degradation level of the crossing as it produced the least MSE compared all inception networks. With ConvNet, the defects can be detected about two months before the defects were spotted during the visual inspection.

Chapter 5

Conclusions and Recommendations

5-1 Conclusions

In this thesis, a methodology to estimate the health conditions of crossings using the Convolutional Neural Networks (CNN) via their degradation levels was proposed. Different experiments were performed, and several conclusions could be drawn a follows:

1. Information regarding defects in crossings could be obtained from transient responses induced by the passing trains. However, the transient responses also contain information related to the train characteristics that could affect the estimation. This condition was demonstrated by the capability of a fully shared convolutional network to classify the types of trains based on the responses recorded by the accelerometer that is located close to the wheel-crossing impact. The localization maps produced by the class activation mapping showed that the higher frequency responses, where defects could also be investigated, contributed to the classifications. Additional accelerometers, which are located in different positions of the crossing (further from the wheel-crossing impact) helped to increase types of trains classifications as they give more information that is less affected by dynamical changes due to defects.

2. In the case of classifying the types of trains, the CNN classifier configured with two parallel convolutional networks where one network was used by the accelerometer located closest to the wheel crossing impact, and another network was shared by two other accelerometers located further from the wheel-crossing impact (2S-CNN), achieved the highest accuracy (93.4%). The worst accuracy (62.09%) was achieved by the CNN classifier configured with a full convolutional network fed

with signals only from the accelerometer located closest to the wheel-crossing impact (FSh1-CNN). However, compared to Support Vector Machine (SVM) and K-Nearest Neighbors (K-NN), FSh1-CNN still outperformed both SVM and K-NN classifiers. The best accuracy of SVM and K-NN classifiers was achieved by the quadratic SVM fed with wavelet entropy (59.54%). SVM and K-NN classifiers are not suitable to perform classification tasks based on linearly inseparable inputs such as wavelet coefficients.

3. The localization maps obtained by using the weighted sum of the last feature maps could be used to locate areas of wavelets coefficients indicating the defects, where the indication of defects was visible from the frequency highlighted by the localization maps ($2.5 - 3$ kHz). Moreover, regression tests to obtain the localization maps were better when performed with a full length of transient responses compared to only using the peak signals. This conclusion was demonstrated by the deviation of the result from the regression test with full transient responses was smaller compared to the one from the regression test using only peak signals.

4. Determining the health condition of the crossing can be done by estimating the degradation level of the crossing. The best CNN architecture for this task was a fully connected convolutional network (ConvNet). Compared to all inception networks, this configuration had the lowest values of Mean Squared Error (MSE) compared to all architectures utilizing inception networks. It means that ConvNet had less risk of triggering false alarms or causing miss detection of failures compared to all Inceptions.

5. It could be concluded from this thesis that the proposed method that uses CNN algorithms could identify symptoms of crossing defects based on the acceleration data before they are visible. In the case study of a crossing used for evaluating the proposed methodology, the symptoms could be identified about two months before the defects were visible.

5-2 Recommendations

Future work will focus on the validation of the methodology and revision of the model. The following activities are recommended as the follow up of this thesis:

Methodology validation

- Build a prototype that could automatically perform measurements to gather more data. The current made use of transient responses generated by about 5000 wheels where only about 500 of them were used for validation (more than 4000 of them were necessary for building the model). For a better validation result, a higher number of data is needed. Additionally, more sensors such as high-definition cameras can also be incorporated as the prototype. Thus, the evolution of defects can be confirmed visually.

- Use the same methodology to estimate the health condition of other crossings. This way, the result of this thesis could be validated, whether similar defects found on other crossings could be identified from the same frequency modes or not.
- The methodology proposed in this thesis was designed to be applicable for a wide range of applications, same steps of the methodology can be used to perform similar tasks such as detecting the incoming defects from axle-box accelerations, strain gauges, or even camera. The result can be utilized to improve this methodology so that it is applicable for wider applications.

Model revision

- This thesis utilized CNN algortihms for feature extraction and degradation level estimation. As new CNN architectures outperforming the existing architectures keep being invented, it is necessary to update the model with the new architectures in order to increase the performance of the methodology.

Appendix A

Support Vector Machine (SVM) and K-Nearest Neighbors (K-NN)

A-1 Support Vector Machine (SVM)

SVM is used in [55] to classify the faults condition of a steam turbine generator. To understand how SVM works, Let $S = \{(x_k, y_k), k = 1, 2, ..., n\}$ is a linearly separable training data, where x_k is the input feature space, and $y_k \in$ {-1, +1} is the class label. SVM seeks a hyperplane $y_H(x) = w_1^T x + b_1$ to divide the data into two classes. The parameter w_1 is the normal vector to the hyperplane, and b_1 is the distance from the origin to the hyperplane. The hyperplane has constraints that $w^T x_k + b \geq 1$ when $y(x_k) = 1$, and $w^T x_k + b \leq$ -1 when $y(x_k) =$ -1, where w is the unit normal vector of the best hyperplane and b is the constant of the best hyperplane. The resulting best hyperplane is illustrated in Figure A-1 (highlighted by a green box).

A-2 K-Nearest Neighbors (K-NN)

K-NN is widely used for nonparametric pattern classification with very little knowledge about the distribution of the data. In [56], it was used for determining health conditions of an induction motor. The K-nearest Neighbor works by comparing given test data with the training data and classify the class of the test data based on the nearest Euclidian distance between the test and training data. The Euclidian distance is defined as $\sqrt{\sum_{i=1}^{n} (m_{1i} - m_{2i})^2}$, where $m_1 = (m_{11}, m_{12}, ..., m_{1n})$ and $m_2 = (m_{21}, m_{22}, ..., m_{2n})$. The k nearest neighbors are used to determine the class of the test data for a better result. The class which has more members in the nearest neighbors becomes the class of the test data. Figure A-2 illustrates how the membership of a data is decided based on its 5-nearest neighbors.

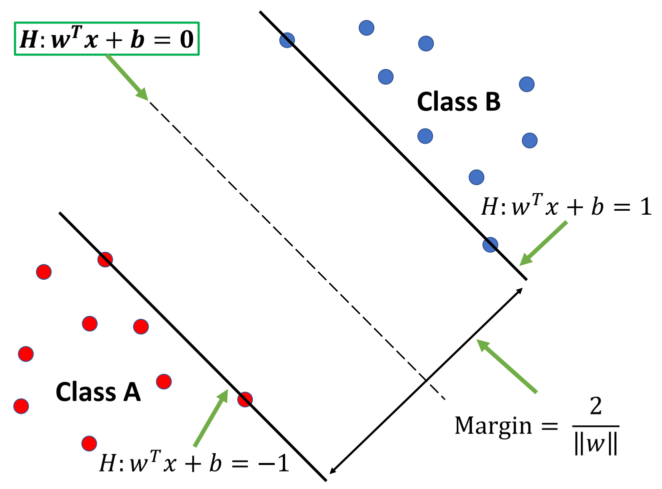


Figure A-1: The hyperplane of SVM was constructed by seeking a hyperplane that can divide two classes with the most possible biggest distance (margin).

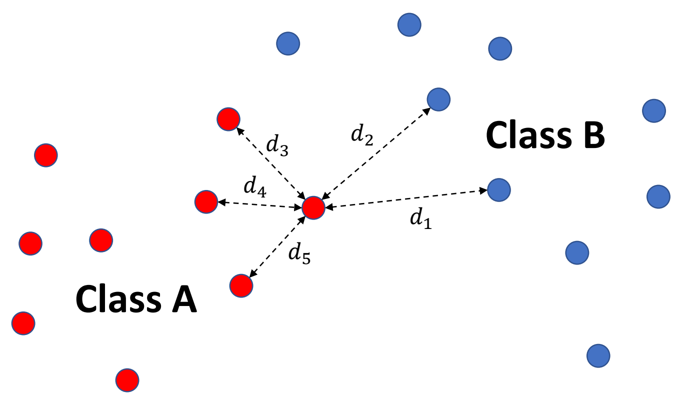


Figure A-2: Illustration of a K-NN classifier deciding the membership of a data using its 5-nearest neighbors. The class of the new member is decided based on a class with the biggest number of members.

Appendix B

Details of Convolutional Neural Networks (CNN) Architectures

Table B-1 contains detail architectures of CNN classifiers used to predict the types of trains. Table B-2, Table B-3, Table B-4, and Table B-5 contain detail architectures of CNN regressors utilized to estimate the degradation level of the crossing.

Table B-1: Details of CNN classifiers used for the types of trains classification.

FSh1-CNN	**FSh2-CNN**	**FS-CNN**			**2S-CNN**	
Input (Accelerometer 1)	Input (Accelerometer 1, 2, 4)	Input (Accelerometer 1)	Input (Accelerometer 2)	Input (Accelerometer 3)	Input (Accelerometer 1)	Input (Accelerometer 2, 4)
zero_padding	zero_padding	zero_padding	zero_padding	zero_padding	zero_padding	zero_padding
8 conv (2,2)	24 conv (2,2)	8 conv (2,2)	8 conv (2,2)	8 conv (2,2)	8 conv (2,2)	16 conv (2,2)
8 conv (2,2)	24 conv (2,2)	8 conv (2,2)	8 conv (2,2)	8 conv (2,2)	8 conv (2,2)	16 conv (2,2)
8 max_pool (2,4)	24 max_pool (2,4)	8 max_pool (2,4)	8 max_pool (2,4)	8 max_pool (2,4)	8 max_pool (2,4)	16 max_pool (2,4)
zero_padding	zero_padding	zero_padding	zero_padding	zero_padding	zero_padding	zero_padding
16 conv (2,2)	48 conv (2,2)	16 conv (2,2)	16 conv (2,2)	16 conv (2,2)	16 conv (2,2)	32 conv (2,2)
16 conv (2,2)	48 conv (2,2)	16 conv (2,2)	16 conv (2,2)	16 conv (2,2)	16 conv (2,2)	32 conv (2,2)
16 max_pool (2,4)	48 max_pool (2,4)	16 max_pool (2,4)	16 max_pool (2,4)	16 max_pool (2,4)	16 max_pool (2,4)	32 max_pool (2,4)
zero_padding	zero_padding	zero_padding	zero_padding	zero_padding	zero_padding	zero_padding
32 conv (2,2)	96 conv (2,2)	32 conv (2,2)	32 conv (2,2)	32 conv (2,2)	32 conv (2,2)	64 conv (2,2)
32 conv (2,2)	96 conv (2,2)	32 conv (2,2)	32 conv (2,2)	32 conv (2,2)	32 conv (2,2)	64 conv (2,2)
32 max_pool (2,4)	96 max_pool (2,4)	32 max_pool (2,4)	32 max_pool (2,4)	32 max_pool (2,4)	32 max_pool (2,4)	64 max_pool (2,4)
64 conv (2,2)	192 conv (2,2)	64 conv (2,2)	64 conv (2,2)	64 conv (2,2)	64 conv (2,2)	128 conv (2,2)
64 conv (2,2)	192 conv (2,2)	64 conv (2,2)	64 conv (2,2)	64 conv (2,2)	64 conv (2,2)	128 conv (2,2)
64 max_pool (2,2)	192 max_pool (2,2)	64 max_pool (2,2)	64 max_pool (2,2)	64 max_pool (2,2)	64 max_pool (2,2)	128 max_pool (2,2)
flatten (1152)	flatten (3456)	flatten (1152)	flatten (1152)	flatten (1152)	flatten (1152)	flatten (2304)
	‘	concatenate (3456)			concatenate (3456)	
fc (1152)	fc (3456)	fc (3456)			fc (3456)	
fc (1024)	fc (2048)	fc (2048)			fc (2048)	
	fc (2048)	fc (2048)			fc (2048)	
	fc (1000)	fc (1000)			fc (1000)	
fc (4)	fc (4)	fc (4)			fc (4)	

Table B-2: Details of CNN architectures (ConvNet) with various depths of convolutional layers.

ConvNet		ConvNet D4		ConvNet D3	
input_2 (defect representative features)	input_1 (input representative features)	input_2 (defect representative features)	input_1 (input representative features)	input_2 (defect representative features)	input_1 (input representative features)
zero_padding	fc (4)	zero_padding	fc (4)	zero_padding	fc (4)
6 conv (2,2)	fc (4)	6 conv (2,2)	fc (4)	6 conv (2,2)	fc (4)
6 conv (2,2)		6 conv (2,2)		6 conv (2,2)	
6 conv (2, 4), stride(2,4)		6 conv (2, 4), stride(2,4)		6 conv (2, 4), stride(2,4)	
zero_padding		zero_padding		zero_padding	
12 conv (2,2)		12 conv (2,2)		12 conv (2,2)	
12 conv (2,2)		12 conv (2,2)		12 conv (2,2)	
12 conv (1, 4), stride(1,4)		12 conv (1, 4), stride(1,4)		12 conv (1, 4), stride(1,4)	
zero_padding		zero_padding		zero_padding	
24 conv (2,2)		24 conv (2,2)		24 conv (2,2)	
24 conv (2,2)		24 conv (2,2)		24 conv (2,2)	
24 conv (1, 4), stride(1,4)		24 conv (1, 4), stride(1,4)		24 conv (1, 4), stride(1,4)	
zero_padding		zero_padding			
48 conv (2,2)		48 conv (2,2)			
48 conv (2,2)		48 conv (2,2)			
48 conv (1, 4), stride (1,4)		48 conv (1, 4), stride (1,4)			
zero_padding					
96 conv (2,2)					
96 conv (2,2)					
96 conv (1, 2), stride (1, 2)					
flatten (1154)		flatten (1154)		flatten (2304)	
concatenate (1156)		concatenate (1156)		concatenate (2308)	
fc (1156)		fc (1156)		fc (2308)	
fc (1156)		fc (1156)		fc (2308)	
fc (1)		fc (1)		fc (1)	

Table B-3: Details of CNN architectures (ConvNet) with different number of convolutional filters.

ConvNet		**ConvNet F4**		**ConvNet F8**	
input_2 (defect representative features)	input_1 (input representative features)	input_2 (defect representative features)	input_1 (input representative features)	input_2 (defect representative features)	input_1 (input representative features)
zero_padding	fc (4)	zero_padding	fc (4)	zero_padding	fc (4)
6 conv (2,2)	fc (4)	4 conv (2,2)	fc (4)	8 conv (2,2)	fc (4)
6 conv (2,2)		4 conv (2,2)		8 conv (2,2)	
6 conv (2, 4), stride(2,4)		4 conv (2, 4), stride(2,4)		8 conv (2, 4), stride(2,4)	
zero_padding		zero_padding		zero_padding	
12 conv (2,2)		8 conv (2,2)		16 conv (2,2)	
12 conv (2,2)		8 conv (2,2)		16 conv (2,2)	
12 conv (1, 4), stride(1,4)		8 conv (1, 4), stride(1,4)		16 conv (1, 4), stride(1,4)	
zero_padding		zero_padding		zero_padding	
24 conv (2,2)		16 conv (2,2)		32 conv (2,2)	
24 conv (2,2)		16 conv (2,2)		32 conv (2,2)	
24 conv (1, 4), stride(1,4)		16 conv (1, 4), stride(1,4)		32 conv (1, 4), stride(1,4)	
zero_padding		zero_padding		zero_padding	
48 conv (2,2)		32 conv (2,2)		64 conv (2,2)	
48 conv (2,2)		32 conv (2,2)		64 conv (2,2)	
48 conv (1, 4), stride (1,4)		32 conv (1, 4), stride (1,4)		64 conv (1, 4), stride (1,4)	
zero_padding		zero_padding		zero_padding	
96 conv (2,2)		64 conv (2,2)		128 conv (2,2)	
96 conv (2,2)		64 conv (2,2)		128 conv (2,2)	
96 conv (1, 2), stride (1, 2)		64 conv (1, 2), stride (1, 2)		128 conv (1, 2), stride (1, 2)	
flatten (1152)		flatten (768)		flatten (1536)	
concatenate (1156)		concatenate (772)		concatenate (1540)	
fc (1156)		fc (772)		fc (1540)	
fc (1156)		fc (772)		fc (1540)	
fc (1)		fc (1)		fc (1)	

Table B-4: Details of CNN architectures with one inception module (average-pool layers or max-pool layers).

Inception Conv + Avg			**Inception Conv + Max**		
input_2 (defects representative features)		input_1 (input representative features)	input_2 (defects representative features)		input_1 (input representative features)
zero_padding	2 avg_pool (2,4)	fc (4)	zero_padding	2 max_pool (2,4)	fc (4)
4 conv (2,2)		fc (4)	4 conv (2,2)		fc (4)
4 conv (2,2)			4 conv (2,2)		
4 conv (2,4)			4 conv (2,4)		
6 concatenate (6,256)			6 concatenate (6,256)		
zero_padding	6 avg_pool (1,4)		zero_padding	6 max_pool (1,4)	
12 conv (2,2)			12 conv (2,2)		
12 conv (2,2)			12 conv (2,2)		
12 conv (1,4)			12 conv (1,4)		
18 concatenate (6,64)			18 concatenate (6,64)		
zero_padding	18 avg_pool (1,4)		zero_padding	18 max_pool (1,4)	
36 conv (2,2)			36 conv (2,2)		
36 conv (2,2)			36 conv (2,2)		
36 conv (1,4)			36 conv (1,4)		
54 concatenate (6,16)			54 concatenate (6,16)		
zero_padding	54 avg_pool (1,4)		zero_padding	54 max_pool (1,4)	
108 conv (2,2)			108 conv (2,2)		
108 conv (2,2)			108 conv (2,2)		
108 conv (1,4)			108 conv (1,4)		
162 concatenate (6,4)			162 concatenate (6,4)		
zero_padding	162 avg_pool (1,4)		zero_padding	162 max_pool (1,4)	
324 conv (2,2)			324 conv (2,2)		
324 conv (2,2)			324 conv (2,2)		
324 conv (1,4)			324 conv (1,4)		
486 concatenate (6,1)			486 concatenate (6,1)		
flatten (2916)			flatten (2916)		
concatenate (2920)			concatenate (2920)		
fc (2920)			fc (2920)		
fc (2048)			fc (2048)		
fc (1)			fc (1)		

Table B-5: Details of CNN architectures with two inception module (average-pool layers and max-pool layers).

Inception Conv + Avg + Max			
input_2 (defects representative features)			input_1 (input representative features)
zero_padding	2 avg_pool (2,4)	2 max_pool (2,4)	fc (4)
2 conv (2,2)			fc (4)
2 conv (2,2)			
2 conv (2,4)			
6 concatenate (6,256)			
zero_padding	6 avg_pool (1,4)	6 max_pool (1,4)	
12 conv (2,2)			
12 conv (2,2)			
12 conv (1,4)			
24 concatenate (6,64)			
zero_padding	24 avg_pool (1,4)	24 max_pool (1,4)	
48 conv (2,2)			
48 conv (2,2)			
48 conv (1,4)			
96 concatenate (6,16)			
zero_padding	96 avg_pool (1,4)	96 max_pool (1,4)	
192 conv (2,2)			
192 conv (2,2)			
192 conv (1,4)			
384 concatenate (6,4)			
zero_padding	384 avg_pool (1,4)	384 max_pool (1,4)	
768 conv (2,2)			
768 conv (2,2)			
768 conv (1,4)			
1536 concatenate (6,1)			
flatten (9216)			
concatenate (9220)			
fc (9220)			
fc (2048)			
fc (1)			

Bibliography

[1] W. Zwanenburg, "Degradation processes of switches & crossings," in *2006 IET International Conference On Railway Condition Monitoring*, (Birmingham, UK), pp. 115–119, 29-30 Nov 2006.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] Z. Chen, C. Li, and R. Sanchez, "Gearbox fault identification and classification with convolutional neural networks," *Shock and Vibration*, vol. 2015, no. 5, pp. 10–21, 2015.

[4] A. Subasi, "EEG signal classification using wavelet feature extraction and a mixture of expert model," *Expert Systems with Applications*, vol. 32, no. 4, pp. 1084 – 1093, 2007.

[5] I. Genc, C. Guzelis, and I. Goknar, "Classification of acoustical alarm signals with CNN using wavelet transformation," in *1996 Fourth IEEE International Workshop on Cellular Neural Networks and their Applications Proceedings (CNNA-96)*, pp. 375–379, Jun 1996.

[6] V. Meruane and J. Mahu, "Real-time structural damage assessment using artificial neural networks and antiresonant frequencies," *Shock and Vibration*, vol. 2014, February 2014.

[7] C. Esveld, *Modern railway track*. Delft, South Holland: MRT Productions, 2nd ed., 2010.

[8] "Herzstück und Radlenker einer Eisenbahnweiche." https://commons.wikimedia.org/wiki/File:Eisenbahnweiche_detail.jpg, 2004.

[9] "Operational failure modes of switches and crossings." http://www.capacity4rail.eu/IMG/pdf/c4r_-_d131_-_operational_failure_modes_of_scs_public_.pdf, 2015.

[10] J. Urda, J. Mandulab, and A. Urdov, "Degradation model of wear frogs switches on the track Z̃SR," *Procedia Engineering*, vol. 111, no. 122, pp. 803–806, 2015.

[11] D. Rama and J. D. Andrews, "A reliability analysis of railway switches," *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, vol. 227, no. 4, pp. 344–363, 2013.

[12] A. Johansson, B. Palsson, M. Ekh, J. Nielsen, M. Ander, J. Brouzoulis, and E. Kassa, "Simulation of wheel-rail contact and damage in switches & crossings," *Wear*, vol. 271, no. 1-2, pp. 472–481, 2011.

[13] L. Xin, V. Markine, and I. Shevtsov, "Numerical procedure for fatigue life prediction for railway turnout crossings using explicit finite element approach," *Wear*, vol. 366, no. 18, pp. 167–179, 2016.

[14] Z. Wei, C. Shen, Z. Li, and R. Dollevoet, "Wheel–rail impact at crossings: Relating dynamic frictional contact to degradation," *Journal of Computational and Nonlinear Dynamics*, vol. 12, p. 041016, feb 2017.

[15] NeTIRail-INFRA, "Cost/benefit data and application methodology for lean in railway S&C." http://netirail.eu/IMG/pdf/netirail-wp2_d2_3_v0_6_final_public.pdf, 2016.

[16] M. Oregui, Z. Li, and R. Dollevoet, "Identification of characteristic frequencies of damaged railway tracks using field hammer test measurements," *Mechanical Systems and Signal Processing*, vol. 54-55, no. Supplement C, pp. 224 – 242, 2015.

[17] M. Oregui, M. Molodova, A. Núñez, R. Dollevoet, and Z. Li, "Experimental investigation into the condition of insulated rail joints by impact excitation," *Experimental Mechanics*, vol. 55, pp. 1597–1612, jul 2015.

[18] Z. Wei, A. Boogaard, A. Núñez, and R. Dollevoet, "An integrated approach for characterizing the dynamic behaviour of wheel-rail interaction at crossings," *Under review*, 2017.

[19] G. Lederman, S. Chen, J. Garrett, J. Kovačević, H. Y. Noh, and J. Bielak, "Track-monitoring from the dynamic response of an operational train: A sparse method," *Mechanical Systems and Signal Processing*, vol. 87, pp. 1–16, 2017.

[20] S. Faghih-Roohi, S. Hajizadeh, A. Núñez, R. Babuska, and B. D. Schutter, "Deep convolutional neural networks for detection of rail surface defects," in *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 2584–2589, July 2016.

[21] P. Salvador, V. Naranjo, R. Insa, and P. Teixeira, “Axlebox accelerations: Their acquisition and time-frequency characterisation for railway track monitoring purposes,” *Measurement*, vol. 82, no. Supplement C, pp. 301 – 312, 2016.

[22] S. Kaewunruen, “Monitoring structural deterioration of railway turnout systems via dynamic wheel/rail interaction,” *Case Studies in Nondestructive Testing and Evaluation*, vol. 1, no. Supplement C, pp. 19 – 24, 2014.

[23] Z. Wei, A. Núñez, Z. Li, and R. Dollevoet, “Evaluating degradation at railway crossings using axle box acceleration measurements,” *Sensors*, vol. 17, no. 10, 2017.

[24] Z. Li, M. Molodova, A. Núñez, and R. Dollevoet, “Improvements in axle box acceleration measurements for the detection of light squats in railway infrastructure,” *IEEE Transactions on Industrial Electronics*, vol. 62, no. 7, pp. 4385–4397, 2015.

[25] G. R. Leone, M. Magrini, D. Moroni, G. Pieri, O. Salvetti, and M. Tampucci, “A smart device for monitoring railway tracks in remote areas,” in *2016 International Workshop on Computational Intelligence for Multimedia Understanding (IWCIM)*, pp. 1–5, Oct 2016.

[26] D. Ribeiro, R. Calçada, J. Ferreira, and T. Martins, “Non-contact measurement of the dynamic displacement of railway bridges using an advanced video-based system,” *Engineering Structures*, vol. 75, pp. 164–180, 2014.

[27] Y. Fan, S. Dixon, R. Edwards, and X. Jian, “Ultrasonic surface wave propagation and interaction with surface defects,” *AIP Conference Proceedings*, vol. 894, no. 1, pp. 894–901, 2007.

[28] P. Rolek, S. Bruni, and M. Carboni, “Condition monitoring of railway axles based on low frequency vibrations,” *International Journal of Fatigue*, vol. 86, pp. 88–97, 2016.

[29] X. Gibert, V. Patel, and R. Chellappa, “Deep multitask learning for railway track inspection,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, pp. 153–164, Jan 2017.

[30] S. Hensel, C. Hasberg, and C. Stiller, “Probabilistic rail vehicle localization with eddy current sensors in topological maps,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1525–1536, 2011.

[31] M. Molodova, M. Oregui, A. Núñez, Z. Li, and R. Dollevoet, “Health condition monitoring of insulated joints based on axle box acceleration measurements,” *Engineering Structures*, vol. 123, no. 18, pp. 225–235, 2016.

[32] P. W. Tse and D. P. Atherton, “Prediction of machine deterioration using vibration based fault trends and recurrent neural networks,” *Journal of Vibration and Acoustics*, vol. 121, no. 3, p. 355, 1999.

[33] R. Tavares de Freitas and S. Kaewunruen, “Life cycle cost evaluation of noise and vibration control methods at urban railway turnouts,” *Environments*, vol. 3, no. 4, p. 34, 2016.

[34] B. A. Pålsson and J. C. Nielsen, “Dynamic vehicle-track interaction in switches and crossings and the influence of rail pad stiffness-field measurements and validation of a simulation model,” *Vehicle System Dynamics*, vol. 53, no. 6, pp. 734–755, 2015.

[35] C. Guo, Y. Yang, H. Pan, T. Li, S. Member, and W. Jin, “Fault analysis of high speed train with DBN hierarchical ensemble,” *2016 International Joint Conference on Neural Networks (IJCNN)*, no. 2015, pp. 2552–2559, 2016.

[36] G. E. Carr and M. D. Chapetti, “On the detection threshold for fatigue cracks in welded steel beams using vibration analysis,” *International Journal of Fatigue*, vol. 33, no. 4, pp. 642–648, 2011.

[37] M. Beyreuther, R. Barsch, L. Krischer, T. Megies, Y. Behr, and J. Wassermann, “Obspy: A python toolbox for seismology,” *Seismological Research Letters*, vol. 81, no. 3, pp. 530–533, 2010.

[38] M. Kristeková, J. Kristek, P. Moczo, and S. M. Day, “Misfit criteria for quantitative comparison of seismograms,” *Bulletin of the Seismological Society of America*, vol. 96, no. 5, pp. 1836–1850, 2006.

[39] R. Wald, T. Khoshgoftaar, and J. C. Sloan, “Fourier transforms for vibration analysis: A review and case study,” *2011 IEEE International Conference on Information Reuse & Integration*, pp. 366–371, 2011.

[40] P. Bentley and J. McDonnell, “Wavelet transforms: an introduction,” *Electronics & Communications Engineering Journal*, vol. 6, no. 4, p. 175, 1994.

[41] H. Zheng-you, C. Xiaoqing, and L. Guoming, “Wavelet entropy measure definition and its application for transmission line fault detection and identification,” *2006 International Conference on Power System Technology*, pp. 1–6, 2006.

[42] Z. YanPing, H. ShuHong, H. JingHong, S. Tao, and L. Wei, “Continuous wavelet grey moment approach for vibration analysis of rotating machinery,” *Mechanical Systems and Signal Processing*, vol. 20, no. 5, pp. 1202–1220, 2006.

[43] A. Phinyomark, A. Nuidod, P. Phukpattaranont, and C. Limsakul, “Feature extraction and reduction of wavelet transform coefficients for EMG pattern classification,” *Elektronika ir Elektrotechnika*, vol. 122, no. 6, pp. 27–32, 2012.

[44] E. Berlin and K. V. Laerhoven, “Sensor networks for railway monitoring: Detecting trains from their distributed vibration footprints,” in *2013 IEEE International Conference on Distributed Computing in Sensor Systems*, (Massachusetts, USA), pp. 80–87, 21-23 May 2013.

[45] D. Yu and L. Deng, *Automatic speech recognition: A deep learning approach*, ch. Deep Neural Networks, pp. 51–76. London: Springer, 1 ed., 2014.

[46] V. Meruane, "Online sequential extreme learning machine for vibration-based damage assessment using transmissibility data," *Journal of Computing in Civil Engineering*, vol. 30, no. 3, pp. 1–10, 2016.

[47] G. Krummenacher, C. S. Ong, S. Koller, S. Kobayashi, and J. M. Buhmann, "Wheel defect detection with machine learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, no. 99, pp. 1–12, 2017.

[48] A. Jamshidi, S. Faghih-Roohi, S. Hajizadeh, A. Núñez, R. Babuska, R. Dollevoet, Z. Li, and B. D. Schutter, "A big data analysis approach for rail failure risk assessment," *Risk Analysis*, vol. 37, pp. 1495–1507, may 2017.

[49] J. Chen, Z. Liu, H. Wang, A. Núñez, and Z. Han, "Automatic defect detection of fasteners on the catenary support device using deep convolutional neural network," *IEEE Transactions on Instrumentation and Measurement*, vol. PP, no. 99, pp. 1–13, 2017.

[50] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[51] E. Kassa and J. Nielsen, "Dynamic interaction between train and railway turnout: full-scale field test and validation of simulation models," *Vehicle System Dynamics*, vol. 46, no. sup1, pp. 521–534, 2008.

[52] S. Suhairy, *Prediction of ground vibration from railways*. Boras: Swedish National Testing and Research Institute (SP), 2000.

[53] H. Azami, K. Mohammadi, and B. Bozorgtabar, "An improved signal segmentation using moving average and savitzky-golay filter," *Journal of Signal and Information Processing*, vol. 03, no. 01, pp. 39–44, 2012.

[54] "Wavelet denoising." https://nl.mathworks.com/help/wavelet/ug/wavelet-denoising.html, 2017.

[55] H. Sun and Y. Huang, "Support vector machine for vibration fault classification of steam turbine-generator sets," *Procedia Engineering*, vol. 24, no. 8, pp. 38–42, 2011.

[56] S. Samanta, J. Bera, and G. Sarkar, "K-nearest neighbor based fault diagnosis system for induction motor," in *2016 2nd International Conference on Control, Instrumentation, Energy Communication (CIEC)*, (Kolkata, India), pp. 304–308, 28-30 Jan 2016.

[57] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[58] R. Kotikalapudi and contributors, “keras-vis.” https://github.com/raghakot/keras-vis, 2017.

[59] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.

[60] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, June 2015.

[61] F. Chollet and contributors, “Keras.” https://github.com/fchollet/keras, 2017.

[62] NS, “Sprinter (sgmm).” https://www.ns.nl/binaries/_ht_1502695321726/content/assets/ns-en/about-ns/2017/sgmm.pdf, 2017.

[63] NS, “Intercity (virm).” https://www.ns.nl/binaries/_ht_1502695324931/content/assets/ns-en/about-ns/2017/virm.pdf, 2017.

[64] W. Zhai, Q. Wang, Z. Lu, and X. Wu, “Dynamic effects of vehicles on tracks in the case of raising train speeds,” *Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit*, vol. 215, no. 2, pp. 125–135, 2001.

Glossary

List of Acronyms

ABA	Axle Box Acceleration
CNN	Convolutional Neural Networks
CWT	Continuous Wavelet Transform
EEMCS	Electrical Engineering, Mathematics and Computer Science
FFT	Fast Fourier Transform
FRF	Frequency Response Function
K-NN	K-Nearest Neighbors
MSE	Mean Squared Error
S&C	Switches and Crossings
SVM	Support Vector Machine