Quantifying uncertainty in structural condition with Bayesian deep learning

A study on the Z-24 bridge benchmark

DAVÍD STEINAR ÁSGRÍMSSON
Quantifying uncertainty in structural condition with Bayesian deep learning

DAVÍÐ STEINAR ÁSGRÍMSSON

Master in Machine Learning
Date: April 27, 2019
Supervisor: Giampiero Salvi
Co-supervisor: Raid Karoumi, Ignacio Gonzalez
Examiner: Danica Kragic
Swedish title: Kvantifiering av osäkerhet i strukturella tillstånd med Bayesiansk djupinlära

School of Electrical Engineering and Computer Science
Abstract

A machine learning approach to damage detection is presented for a bridge structural health monitoring system, validated on the renowned Z-24 bridge benchmark dataset where a sensor instrumented, three-span bridge was realistically damaged in stages. A Bayesian autoencoder neural network is trained to reconstruct raw sensor data sequences, with uncertainty bounds in prediction. The reconstruction error is then compared with a healthy-state error distribution and the sequence determined to come from a healthy state or not. Several realistic damage stages were successfully detected, making this a viable approach in a data-based monitoring system of an operational bridge. This is a fully operational, machine learning based bridge damage detection system, that is learned directly from raw sensor data.

Keywords: Bayesian Deep Learning, Autoencoders, Bridge Structural Health Monitoring, Bridge Damage Detection
Sammanfattning

Contents

1 Introduction 1
   1.1 Aim and scope of thesis 2
   1.2 Social relevance 3

2 Background 4
   2.1 Structural health monitoring 4
   2.2 Machine learning 8

3 Method 17
   3.1 Z-24 benchmark 20
   3.2 Model assumptions 21
   3.3 Model setup 27
   3.4 Damage detection algorithm 31

4 Result 35

5 Discussion 40
   5.1 Detected damage cases 41
   5.2 Damage sensitivity 42
   5.3 Ethical implications 44

6 Conclusion 45
   6.1 Case for real life use 45
Chapter 1

Introduction

Structural health monitoring is a vast and mature field within engineering. Assessing the health of a structure is of vital importance when dealing with critical infrastructure. It turns out that the assessment of structural health is often non-trivial and beyond the scope of classical engineering.

For the layman, it would be easy to assume that bridges are inherently safe, immovable and without fault. The reality is that each bridge is uniquely designed for an assumed workload, and for a specified lifetime. Building materials are ever changing and evolving. The construction itself can present many challenges. Weather and winds have a huge factor in the operation and structural soundness of a bridge. This can result in bridges being used beyond their initial engineering design.

The interplay of these variables presents uncertainty. Uncertainty of condition, uncertainty of state. Regular maintenance and inspection is the traditional approach to reduce this uncertainty. One of the goals of structural health monitoring is to address and quantify this uncertainty in an automatic way.

There are two main approaches for structural health monitoring of bridges, a model based approach and a data based approach. In both cases, sensors if different kinds are attached to the structure to assess its state. In a model based approach, a finite element analysis is done specifically for each bridge which is calibrated with the sensor data. This analysis produces a baseline for values of displacement, strain and vibration at each point. The real sensor measurements are then compared to the calibrated finite element model, and damage is
assessed. In a data based approach, only the sensor data is used for damage assessment. Sensor data is collected for a certain period of time, considered to be the structurally healthy baseline. Deviations from the baseline is assessed as structural damage.

The two fields of structural engineering and machine learning seem to be far removed but their roads cross in the data based approach of structural health monitoring. This thesis is made with the viewpoint of machine learning, to present a state of the art approach for structural engineering applications of structural health monitoring.

1.1 Aim and scope of thesis

This thesis presents a Bayesian deep learning model framework for assessing the structural health of a real, sensor instrumented bridge in the context of a data based structural health monitoring system. The aim of this thesis is to develop a strong model framework for detecting structural damage through vibration sensor data in bridges. The framework should be viable for real life usage, in implementation and from a computational viewpoint.

The results of the thesis should be of interest to researchers in the field of structural health monitoring for sensor instrumented bridges, researchers in Bayesian deep learning and researchers in smart infrastructure. In a broader perspective, the results could be of interest to government transport administration in charge of building, operating and maintaining public bridges, roads and railways.

Certain aspects of structural health monitoring are beyond the scope of this thesis. We will not go into other challenges of implementing a bridge structural health monitoring system with regards to installing sensors or other operational issues regarding sensors, data collection and storage. The machine learning framework will not specifically address the case of faulty sensor readings. The presented framework will not be thought of as a replacement for regular bridge inspection, but as an aid.
1.2 Social relevance

Society depends on safe and reliable infrastructure. Advancements in structural health monitoring using machine learning methods would benefit national infrastructure as a whole. Reduction in maintenance and operational cost or the extension of structural lifetime could potentially save enormous amounts of government spending on infrastructure.
Chapter 2

Background

2.1 Structural health monitoring

Farrar and Worden [2012] define axioms of structural health monitoring. In short, all materials have flaws, at some scale. For bridge structural health monitoring, bridges are instrumented with sensors that measure vibration, strains, environmental values and sometimes more. With this sensor data the goal is to assess the structural health of a bridge. It is important to note, that sensors do not necessarily measure damage. Features are extracted from the sensor data to assess damage, and this feature extraction is done with advanced signal processing or machine learning methods.

There are many methods of detecting damage, and this is a highly researched subject. One aspect that is in common with all of them is a comparison of current state to a baseline state. The methods can be categorized into two main categories, a model based approach and a data based approach.

In a model based approach, a finite element analysis is done specifically for each bridge. The finite element method divides the model bridge into discrete elements that are easily formulated and understood, which are calibrated with sensor data. This analysis produces a baseline for values of displacement, strain and vibration at each point. The real sensor measurements are then compared to the finite element values, and damage is assessed. See figure 2.1 for a simple finite element model of the Z-24 bridge, where a modal analysis was performed.

In a data based approach, only the sensor data is used for damage assessment. Sensor data is collected for a certain period of time, con-
EXPERIMENTAL VALIDATION

In the visualisation of the bending stiffness the inaccurate zones due to zero by zero division (see Eqn.1) are omitted. For the first mode four inaccurate zones are present: at the bridge abutments and at two points of the central span, close to the bridge piers. It turned out that higher modes with more inaccurate zones give unsatisfactory EI estimations. The direct stiffness calculations for damage scenarios are compared with the scenarios without any damage. Also the bending stiffness from the ANSYS finite element model of the undamaged bridge is given.

Firstly the 2nd reference test carried out after cutting the Koppigen pier and installing the steel plates, and the settlement scenarios are compared. The dynamic bending stiffness distribution for the 2nd reference test and the scenario with 40mm settlement (Fig. 6a) corresponds well along the length of the bridge, except at the inner side of the pier which was subjected to the settlement (Koppigen pier at 46.7m). Curvatures are rather small in the side spans, which causes numerical difficulties to calculate EI. Also the higher stiffness of the girder box beam at the girder-pier connections due to the increased thickness of the lower slabs, is clearly detectable from the direct stiffness determination. Another observation is that the Koppigen pier seems to be stiffer than the Utzenstorf pier.

Figure 2.1: A finite element model of the Z-24 bridge showing the first vibrational mode shape. This was obtained by first creating a finite model of the bridge, and then updating the parameters using the real recorded vibrations. Figure taken from Peeters and De Roeck 1998.

Figure 2.2: Vibrational modes of an ideal string, to give an intuitive explanation of modal analysis of a bridge.

cidered to be the structurally healthy baseline. Deviations from the baseline is assessed as structural damage. This thesis builds upon the work of Gonzalez and Karoumi [2015] Neves et al. [2017], where a neural network is trained on the undamaged sensor data, and large deviations from true model predictions is interpreted as damage.
One of the biggest challenges of structural health monitoring is the natural environmental response. Mode shape and natural frequency methods have been extensively researched. Bart Peeters and Guido De Roeck (2001) showed significant changes in a bridge’s natural frequency in freezing temperatures, making a modal approach unstable. Seasonal changes in temperature can result in nonlinear changes in natural frequencies, especially in freezing conditions (Gonzalez, Ulker-Kaustell, and Karoumi 2013). This nonlinearity calls for more complex modeling.

Casas and Moughty (2017) and Das, Saha, and Patro (2016) present an overview of developments in vibration-based methods. They can address some weaknesses of traditional methods and be more robust to environmental variability and more sensitive to damage. When using such methods in a data-based approach, the need for advanced machine learning methods becomes evident, due to the complexity of the data and the huge amount of data that can be collected with sensors.

Gonzalez and Karoumi (2015) use bridge vibrational data and a specialized bridge weigh in motion system that records the bridge’s load position, magnitude, and speed. A neural network is trained to predict the vibrations at each timestep. A Gaussian process was then used to characterize the prediction errors.

Neves et al. (2017) extends the idea from Gonzalez and Karoumi (2015) without the specialized weigh in motion system. This method is promising, but the machine learning methods can be improved given the enormous strides in the field for the past years. This thesis aims to do just that.

The need for uncertainty in structural health monitoring is very prominent, but not all researchers address this. Simoen, Guido De Roeck, and Lombaert (2015) show the need for uncertainty in a data-based approach. Several Bayesian methods have been proposed, and among the most prominent are Bayesian neural networks and Gaussian processes, Teimouri et al. (2017) make the case for the value of uncertainty measures and present a framework based on Gaussian processes as being more robust than neural networks for damage detection in a composite airfoil structure.

Bayesian deep learning shows great promise to combine the strengths of all the proposed methods for data-based structural health monitoring.

This field is very varied, and there is no default method to detect damage. The use of artificial intelligence for structural health moni-
Monitoring is well known, and has recently shown great promise in many different applications (Gomes et al. 2018). Salehi and Burgueño 2018 present a review of machine learning methods in structural health monitoring showing a vast array of different methods used in the literature with the conclusion that machine learning methods are becoming established as an efficient alternative approach to classical modeling techniques in structural engineering.

The environmental variations present considerable challenges to the field. Gonzalez, Ülker-Kaustell, and Karoumi 2013 showed a considerable increase in bridge stiffness during winter and a elastic modulus of ballast increases by one order of magnitude.

**Operational Evaluation**

One major issue faced by every bridge structural health monitoring system is how to evaluate such a system. Datasets that contain both healthy-state and damaged-state bridge sensor data are extremely rare, with only a handful being available, so evaluation of a real system is very limited. On top of that, each bridge is unique - so a structural health monitoring system will have to be specifically implemented for each bridge.

A common way to tackle this is to create a computerized three-dimensional finite element model of a bridge, and numerically simulate the bridge sensor data. This models the dynamics of the structure. Damage is then introduced by modifying the model structure. See figure 2.1 for such a model of the Z-24 bridge. Another way is to create a physical miniature bridge model, instrument it and induce vibrations. Such a model can then be physically damaged to imitate a real scenario. A strong approach is to implement both, as seen in (Kullaa 2011). Such simulations, in both cases, can never fully capture the complexity of a real life scenario, and are therefore mostly useful as a method proof-of-concept.

The most faithful way to evaluate a bridge structural health monitoring system is to use the system in conjunction with regular bridge inspection and maintenance. If the system predicts damage, it has to be physically verified. If damage is apparent during inspection, the model can be verified.

Sensor validation is another issue, where failing sensors will corrupt the system performance (Yi, Huang, and H.-N. Li 2017).
2.2 Machine learning

Deep neural networks are models comprised of many building blocks of machine learning. To provide more clarity of the methods used, I will go through these building blocks in a unifying framework, inspired by Bishop (2006).

Linear regression is the most basic building block, a linear combination of \( D \) dimensional input variables \( X = x_1, \ldots, x_D \) and weight parameters \( w = w_0, w_1, \ldots, w_D \). A bias term \( x_0 \) is added to the input, to make the input into the model \( D+1 \) dimensional. The output prediction \( \hat{y} \) is on the form of a matrix dot product

\[
\hat{y} = w^T X
\]

Given \( N \) observations of \((X, y)\) pairs, there exists a closed form solution to the \( w \) that minimized the error function. For linear regression, the error function is usually the mean squared error function. The error is usually assumed to be normally distributed with zero mean.

Logistic regression is very similar in form, except that the output prediction is binary. A non-linear activation function \( \sigma(x) \) outputs a value between 0 and 1, that can be interpreted as the probability of a class being 1, as seen in equation (2.2). On the other hand, given \( N \) observations of \((X, y)\) pairs, there is no closed form solution to the weights \( w \). Finding a \( w \) that minimizes chosen error function is a convex optimization problem.

\[
\hat{y} = \sigma(w^T X)
\]

A feed forward neural network is a composite of many logistic regression blocks. The blocks are not observed, and therefore often called hidden. In a single hidden layer feed forward neural network as seen in figure 2.4 on the form of equation (2.3), there are two weight matrices \( w_1, w_2 \) that need to be optimized to minimize the error function.

\[
\hat{y} = w_2 \sigma(w_1^T X)
\]

This optimization problem requires more sophisticated methods. The most common methods are based on back-propagation (Kelley 1960; Bryson and Denham 1962; Rumelhart, G. E. Hinton, and R. J. Williams 1986) using variations of stochastic gradient descent (Sutskever et al. 2013; Kingma and Ba 2014).
Figure 2.3: Comparison of linear regression and logistic regression. Shaded units mean that they are directly observed. The activation function in logistic regression is a sigmoid function $\sigma(x) = \frac{e^x}{e^x + 1}$ plotted inside the shaded unit.

Figure 2.4: Neural network with a single hidden layer and a width of six hidden units with ReLU as an activation function, $\sigma(x) = \max(0, x)$

**Gaussian processes for regression**

As defined by Rasmussen and C. Williams (2006), a Gaussian process is a collection of random variables, any Gaussian process finite number of which have a joint Gaussian distribution.
A Gaussian process can be thought of as a probability distribution over functions, and the real process \( f(x) \) is completely specified by the mean function
\[
m(x) = E[f(x)]
\]
and covariance function
\[
k(x, x') = E[(f(x) - m(x))(f(x') - m(x'))]
\]
so the Gaussian process can be written as
\[
f(x) \sim GP(m(x), k(x, x'))
\]

There is a connection between neural networks and Gaussian processes. A neural network in the limit of a single, infinitely wide hidden layer approximate Gaussian Process (Neal [1996b]) when there is a prior distribution on the weights. This neural network interpretation of a Gaussian process can be seen in figure 2.5.

![Figure 2.5: Gaussian process approximation in the infinitely wide limit of a single hidden layer neural network](image)

The problem with a full GP is the computational complexity, especially for large data sets. For a given training data set \( D \) containing \( N \)
observations, the computational complexity of training is quadratic in the number of observations $N$ because of the inversion of the covariance matrix $k(x, x')$.

Sparse Gaussian processes (Snelson and Ghahramani 2006) are proposed as a method to use Gaussian processes in practice with large data, to reduce computational complexity. Sparse Gaussian processes use a lower rank approximation of the full covariance function. The information from the training data is compressed into pseudo observations, and only those are stored rather than the full data. This method reduces the computational complexity, but introduces hyperparameters that need to be optimized.

**Deep neural networks**

A neural network with many hidden layers is called a deep neural network. A deep neural network can theoretically approximate any continuous function (Leshno et al. 1993). Even a shallow neural network can theoretically do the same (Cybenko 1989). In practice, there is a balance between the width and the depth of the hidden layers. By increasing the width and depth, the neural network can approximate functions of more complexity, but training the network becomes difficult. Recently, focusing on increasing the depth has been shown to be effective (Krizhevsky, Sutskever, and G. E. Hinton 2012; Simonyan and Zisserman 2014; He et al. 2016).

The cascading effect of a deep neural network presents a way to represent multiple levels of abstraction (LeCun, Bengio, and G. Hinton 2015). The model output $\hat{y}$ in equation (2.7) of the deep neural network in figure 2.6 can be interpreted as a composite of simpler functions $g$.

$$\hat{y} = g_4 (g_3 (g_2 (g_1(X)))) \quad (2.7)$$
Autoencoders

An autoencoder is a neural network that is trained to reconstruct a given input, through the constraints of a lower dimensional bottleneck layer. This can be seen as an unsupervised method, since there is no requirement of labeled data. The idea is to reduce the dimensionality of the data to a low dimensional representation (G. E. Hinton and Salakhutdinov 2006). The main features of an autoencoder is the encoder, the code, and the decoder. The encoder maps the input to a low dimensional code, a compact representation of the original input. The decoder then reconstructs the original input from the low dimensional code. This can be seen in figure 2.7.
Bayesian deep learning

For many problems, the ability to represent uncertainty is crucial. Deep learning methods have been wildly successful in certain areas, but they do not output model uncertainty. Bayesian modeling, such as Gaussian processes, takes uncertainty into account in a principled manner. The intersection of these two fields tries to make the best of both worlds.

Neal (1996a) and MacKay (1992) presented this idea, but they were in a way ahead of their time. Neural networks were promising, but their performance did not live up to expectations. It was not until the work of G. E. Hinton, Osindero, and Teh (2006) that deep neural networks as we know them today became feasible. Today, this merging of fields is very actively researched. In this chapter I will go through two important findings, a deep Gaussian process and Monte Carlo Dropout.

A deep Gaussian process (Damianou and Lawrence 2013) uses sparse Gaussian Process as a key approximation in combination with multiple hidden layers of latent variables. Pseudo inputs are created
between layers using Singular Value Decomposition (Golub and Reinsch 1970) and act as a bottleneck input to the next layer. This in effect is a cascade of functions, much in the spirit of equation (2.7), a composite stochastic process. See figure 2.8 for illustration. This model is both deep, and can handle uncertainty in prediction.

Gal and Ghahramani (2016) presented an interpretation of dropout (Srivastava et al. 2014) as approximate Bayesian inference in deep Gaussian processes. This method allows obtaining model uncertainty from any standard neural networks using dropout. Usually, dropout is used during training of neural networks to reduce overfitting, but for prediction no dropout is used. Each unit is dropped with a probability $p$, making the final network more robust. See figure 2.9 for two stochastic forward passes using dropout.

Gal and Ghahramani make a theoretically grounded observation of averaging many stochastic forward passes with dropout, to achieve a prediction with uncertainty. Training a neural network with dropout
can be seen as training a collection of randomly sampled networks that share the same weights. A single neural network then approximates the prediction by averaging. Averaging many dropout forward passes is equivalent to Monte Carlo integration over a variational approximation of a Gaussian process posterior, referred to as MC dropout. The mean of the predictions is simply the predictive mean, and the variance of the predictions is the uncertainty. The method is theoretically grounded, easy to implement and computationally efficient.

Some promising results of this method are seen in image segmentation (Isobe and Arai 2017; Kamnitsas et al. 2017) where uncertainty is critical and in medical image diagnosis (Leibig et al. 2017). One of the biggest challenges of MC dropout is to achieve well-calibrated uncertainty estimates, some extensions have been made (Y. Li and Gal 2017; Gal, Hron, and Kendall 2017) to address this issue.

The model used in this thesis will be comprised of an feedforward autoencoder neural network, made approximate Bayesian by using MC Dropout, so that the autoencoder will output uncertainty in its reconstruction of the input data.
Figure 2.9: Two stochastic forward passes of a deep neural network with dropout, each unit is dropped with a probability of $p = 0.5$. For MC Dropout, many such stochastic passes are averaged to obtain a prediction with uncertainty.
Chapter 3

Method

![Diagram showing training data and healthy state data]

**Figure 3.1: A central theme of this thesis.** The collected healthy-state data used to train a model represents only a part of possible healthy-state data. The predictive error for a test point A close to the training data set should be lower than for a point B far away. Using only high predictive error for anomaly detection would likely falsely flag point B as anomalous. By also using the models predictive uncertainty, the high predictive error for point B could be explained by the high predictive uncertainty. On the other hand, if a test point has high predictive error beyond the uncertainty bounds it is flagged as an anomaly. The method should therefore be able differentiate between points B and C.

The method to detect change in structural condition presented in this thesis builds upon the work of (Gonzalez and Karoumi 2015; Neves et al. 2017). A Bayesian deep neural network is trained on the collected healthy bridge vibrational and environmental sensor data to reconstruct the given healthy-state input data. The autoencoder neural network reconstructs the input data with an uncertainty interval by using MC dropout. The reconstruction error of the unseen healthy-state data is then quantified and a 95th percentile rejection threshold is determined.
If the total sequence reconstruction error of a sequence in an unknown state falls outside the determined rejection threshold, then the sequence is rejected. If enough sequences are rejected, the bridge is determined to be in a damaged state.

The need for uncertainty is explained in figure 3.1. The collected training data for bridge structural health monitoring can not be truly complete in a reasonable time frame, due to potentially large variations in regular environmental variables. Such a complete recording would take many years. Therefore a predictive model is nearly always going to be trained on an limited healthy-state dataset. It is important to note that a healthy-state can still contain old damages even though we consider this base state as undamaged. It is of course not possible to detect damages induced in the bridge before the sensors were installed.

A crucial observation is what happens when the model is faced with unfamiliar input data, depicted as point B and C in figure 3.1. By only relying on model error from true values, it would be easy to characterize large error as anomalous as for point B and C. By including model uncertainty, the reconstruction of point B will have large error but the model should be able to explain the large error by having large uncertainty and therefore not label the large error as anomalous. If the true values fall outside the predictive uncertainty of the model as in point C, the model can not explain the large difference between real and predicted values and therefore rejects the sequence. See figure 3.2c for a depiction of the crucial information predictive uncertainty gives in reconstruction.
Figure 3.2: The crucial information of a model’s predictive uncertainty in anomaly detection.
3.1 Z-24 benchmark

One of the greatest challenges of bridge structural health monitoring is that there are very few datasets that contain data of a bridge in a healthy state and in a damaged state. The Z-24 benchmark (Reynders and Guido De Roeck 2014; Reynders and Roeck 2009) contains sensor recordings of a real full-sized bridge that was monitored for almost a year before being deliberately damaged in a realistic and controlled way. Damage was introduced in 14 steps, each damaging the bridge further in a realistic manner. This is useful to assess the models sensitivity, to see at what stage the model can assess damage.

![Figure 3.3: The dataset visualized. Only the healthy bridge condition data is used to train the models. The best model is picked by performance on the validation data. The errors are quantified on the unseen test data. The true model performance is then assessed on each damage case.](image)

Both vibrations and environmental variables were recorded such as air temperature, soil temperature, humidity, wind speed and car passing. Sensor recordings were made every hour, with one such vibrational recording depicted in figure 3.8. A total of 53 environmental variables were monitored each hour, with a majority being temperature.

![Figure 3.4: Side view of the Z24 bridge, distances in in meters (Reynders and Roeck 2009).](image)
CHAPTER 3. METHOD

2. Civil Engineering Applications

2.70 2.70 14.00 14.00 30.00
KoppigenUtzenstorf
1.10
To Bern 4.50 To Zurich

Figure 1. Side view of the Z24 bridge. Distances are in meters.

Figure 2. Cross section of the girder, showing the locations where the temperature was monitored.

progressive damage tests were alternated with short-term monitoring tests while the continuous monitoring system was still running during these tests.

The Z24 bridge project was unique in the sense that a long-term continuous monitoring test was combined with realistic short-term progressive damage tests. The measurement data have been used for two benchmarks:

• The shaker, ambient, and drop weight vibration data from the third reference measurement on the Z24 bridge (scenario 8, Table 1) were presented as a benchmark study for system identification methods for operational modal analysis at the IMAC XIX conference in 2001 (Section 3).

• The data from the long-term continuous monitoring tests as well as the data from the progressive damage tests were presented as a benchmark study for algorithms for structural health monitoring and damage identification in the framework of the European Cost Action F3 (Sections 4 and 5).

The benchmark data are still publicly available, and some recently developed methods for modal analysis, damage identification, and health monitoring have been tested on the data.

This article is organized as follows. The different tests are described in Section 2. To provide some insight into the dynamic behavior of the bridge, the operational modal analysis results are discussed in Section 3. Sections 4 and 5 contain a literature review of results obtained from the continuous monitoring data and the progressive damage test data, respectively.

2 TEST DESCRIPTIONS

In this section, a brief overview of the performed tests is given. A profound overview is provided by Krämer et al. [2].

2.1 Long-term Continuous Monitoring Test

Since the aim of this test was to quantify the environmental variability of the bridge dynamics, all environmental variables that were considered to be of possible importance for the bridge dynamics have been monitored.

Sensors to measure air temperature, air humidity, rain true or false, wind speed, and wind direction were installed at the bridge, resulting in five sensors for the atmospheric conditions.

A sensor consisting of two inductive loops was installed to detect the presence of vehicles on the bridge.

Since temperature was known to have a key influence on the dynamics of civil engineering structures, the bridge’s thermal state was monitored in detail. At the middle of the three spans, the temperature was monitored.

Figure 3.5: Placement of environmental sensors in a cross section of the Z-24 bridge.

3.2 Model assumptions

"A model is a set of assumptions about the data."

Christopher Bishop

In this section, the most important assumptions of the modeling are stated. Some of them are limiting, and some make the problem more feasible.

1. Vibration sensor data is correlated

Figure 3.6: A sample of recorded vibrations of all eight sensors normalized and plotted on a mutual time axis. Here you can see that the recorded vibrations are heavily correlated, naturally since the bridge is one continuous unit. When there is heavy vibration in one sensor, it is reasonable to assume that other sensors behave similarly.

Since the sensors are all connected to the same continuous bridge, it is reasonable to assume that the sensor recordings are heavily correlated.
When one sensor is detecting high vibration, it is safe to assume that the other sensors are detecting high vibrations too. This becomes very clear in figure 3.6. Since the vibrations are highly correlated, it makes the prediction problem easier than if they were uncorrelated. Due to this assumption, it is important that the input to the model to reconstruct sensor values contains the data of all sensors on the bridge. This input can be seen in figure 3.7.

2. Vibrations are dependent on environmental variables

Environmental variables such as air temperature, soil temperature, humidity, wind speed and car passing have a big impact on the response of a bridge. This fact is detailed in chapter 2.1. It is therefore of crucial importance to include these variables in the input to the predictive model. A neural network is flexible in the choice of input dimensionality, making this task straightforward. For the Z-24 bridge benchmark, 53 environmental variables were measured each hour alongside the vibrations and therefore will be included in the input of the model.

3. Enough information is contained within a short time frame

The chosen neural network model can only take a fixed, limited size input. A reasonable input to the model would be a sliding window of sensor data in the order of seconds. This slicing of the complete data can be seen in figure 3.7 as the vibrational input into the model. This is a limitation to the model since the behaviour could possibly not be captured in such a short time frame, but trends can still be noted. The first two sensors in figure 3.7 are clearly out-of-phase waves.

4. The vibration data is hierarchical

The nature of a deep neural network is in the cascading composition of simpler elements, as seen in equation (2.7). The vibrations of a bridge are the end result of many inter-playing variables such as weather, winds, loads and elements. It is therefore not far fetched that the data is inherently compositional, and a deep neural network would be a good fit for this problem.
Figure 3.7: An example of a sliding window input of vibration sensor data, where the window size is one second.

5. A vibration sensor measures local condition

One of the pillars of bridge structural health monitoring is the localization of damage. To address this, it is assumed that if the real sensor values for a given sensor are far away from the models predictive uncertainty of that particular sensor, the bridge is damaged in the region of the physical location of the sensor.

This assumption holds in many damage scenarios, but for a damage in a main longitudinal load carrying beam can also be seen in sensors located far from the damage location compared to a damage of a secondary load carrying member such as a crossbeam which will only
influence local measurements. This is out of the reach of the suggested approach.

6. A damaged state changes the general response of the bridge.

A damaged bridge generally behaves in a more complex manner than an undamaged bridge. Damage results in more vibration, higher damping and more complex non-linear behaviour. It is therefore assumed that after damage is present, it will continuously affect the behaviour of the bridge. Note that in real life this is not always the case.

Due to this assumption, we can not assess damage from a single prediction. A long, continuous sequence needs to be assessed to determine bridge state.
Figure 3.8: A full ~10 minute recorded vibration sequence from a healthy-state, all 8 vibration sensors.
Figure 3.9: A full ~10 minute recorded vibration sequence from a damaged state, all 8 vibration sensors.
3.3 Model setup

Training

The models autoencoder neural network architecture is depicted in figure 2.7. The model is trained using Dropout (Srivastava et al. 2014), illustrated in figure 2.9. The autoencoder is trained on the healthy state test dataset, as illustrated in figure 3.3, to reconstruct the healthy state bridge sensor data. An example of input to the autoencoder is shown in figure 3.7. The autoencoder reconstructs vibration sensor data from all sensors at once, due to assumption 1. Environmental sensor data is also included in the input due to assumption 2.

Validation

To prevent a biased model being picked, an model architecture search is performed on a validation test set. Multiple models are trained using Random Search for hyperparameter optimization (Bergstra and Bengio 2012). The hyperparameter values of each trained model are randomly sampled from a Uniform distribution; \( x \in \text{Uniform}[\text{low}, \text{high}] \). The Uniform distribution is either continuous or discrete, depending on the nature of the hyperparameter. Hyperparameters are specified in table 3.1.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Range</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Hidden layer width</td>
<td>[256, 512]</td>
<td>Discrete</td>
</tr>
<tr>
<td>Second Hidden layer width</td>
<td>[128, 256]</td>
<td>Discrete</td>
</tr>
<tr>
<td>Z code layer width</td>
<td>[64, 128]</td>
<td>Discrete</td>
</tr>
<tr>
<td>Dropout</td>
<td>[0.05, 0.3]</td>
<td>Continuous</td>
</tr>
<tr>
<td>Learning rate</td>
<td>[0.00005, 0.001]</td>
<td>Continuous</td>
</tr>
<tr>
<td></td>
<td>[Xavier normal, Xavier uniform, Kaiming uniform, Kaiming normal, Orthogonal]</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: Hyperparameter search setup

Adam optimizer (Kingma and Ba 2014) was used. The model ar-
chitecture with the lowest mean squared reconstruction error on the validation set is then chosen as a final model.

**Testing**

The final trained model based from the validation step is used on the *never seen healthy state test set*. This gives an unbiased model performance measure. The reconstruction error on the healthy state test dataset is the most important measure to quantify the normal range of model error when the bridge is in a healthy state. See figure 3.10 of the distribution of healthy-state test set window error of three different reconstruction approaches.

**Determining error threshold**

It is not reasonable that the model can determine if the bridge is in a damaged state by only looking at reconstruction error from a sensor input of a few seconds. A much more natural approach is to look at the total reconstruction error from a long, continuous sequence as illustrated in figure 3.8.

A central idea in this method is therefore to quantify the summed window reconstruction error of the full 10 minute sequences in the health-state test dataset. 5% of the healthy-state sequences are rejected, based on the total sequence reconstruction error. This is to have a baseline of comparison of summed sequence error of a bridge in an unknown state. We know that 5% of never seen, healthy-state sequences are rejected. If a significantly higher portion of sensor window sequences of a bridge in an unknown state are rejected, the bridge is considered to be damaged. The determined threshold can be seen in figure 3.11a, 3.11b, 3.11c and the classification in figure 4.1, 4.2 and 4.3.
Figure 3.10: Distribution of sliding window errors on the healthy-state test set. N = 235000.
Figure 3.11: The distribution of healthy-state testset summed sequence errors, showing the determined 95% rejection threshold for each method.
3.4 Damage detection algorithm

For the damage detection, three algorithms are presented. Note that all three algorithms use the exact same trained model obtained as described in section 3.3. It is therefore reasonable to use all three in a real life system, since they are variations on a theme. All three methods reconstruct a given 10 minute long sensor sequence, figure 3.8, in a succession of sliding window inputs, figure 3.7, with the autoencoder neural network trained on the healthy-state data. The reconstruction error for the whole sequence is summed and compared to the distribution of total sequence error for the healthy-state data, figure 3.11. If the total reconstruction error of the given sequence is over the 95% cutoff rejection threshold, the sequence is determined to be from a damaged-state.

The **standard method** reconstructs the input in one single forward pass of the neural network. The method does not take into consideration any notion of uncertainty. This is similar to figure 3.2a. Pseudocode for this method can be seen in listing 3.1.

The **mean method** uses MC Dropout (Gal and Ghahramani 2016) to stochastically reconstruct the input 100 times, and to use the mean of the 100 reconstructions as the final prediction. This method gives a Maximum A Posteriori estimation. Pseudocode can be seen in listing 3.2.

The **uncertainty method** uses MC Dropout (Gal and Ghahramani 2016) to stochastically reconstruct the input 100 times. True values that fall within 2 standard deviations from the mean of the predictions are then considered to have 0 error. Only values that fall outside the uncertainty range are taken into the reconstruction error. The sequence in figure 3.2b would therefore have 0 reconstruction error, and the sequence in figure 3.2c would only include the reconstruction error where the true values fall outside the 2 standard deviation range of the predictions. The reasoning behind this approach, is to be able to disregard reconstruction error that falls within the range of expected values. This method therefore only reports errors that are considerably different from the true values. Pseudocode can be seen in listing 3.3.
### Sequence reconstruction

total_error = 0

for window in sequence:
    X = predict(window)
    error = mean_squared_error(window, X)
    for e in error:
        # cutoff
        if e > 0.5805748701095581:
            e = 0.5805748701095581
        total_error += sum(error)

### Classification

if total_error > 88.4090571925044:
    classification = 'damaged'
else:
    classification = 'healthy'
### Sequence reconstruction

\[ N = 100 \]

\[ \text{total\_error} = 0 \]

\[ \text{for window in sequence:} \]

\[ X = [] \]

\[ \text{for i in range}(1:N): \]

\[ X[i] = \text{stochastic\_predict}(\text{window}) \]

\[ X\_mean = \text{mean}(X) \]

\[ X\_2std = 2*\text{std}(X) \]

\[ \text{error} = \text{mean\_squared\_error}(\text{window}, X\_mean) \]

\[ \text{for e in error:} \]

\[ \quad \# \text{cutoff} \]

\[ \quad \text{if e} > 0.4070758819580078: \]

\[ \quad \quad e = 0.4070758819580078 \]

\[ \quad \text{total\_error} += \text{sum}(\text{error}) \]

### Classification

\[ \text{if total\_error} > 53.652546491939574: \]

\[ \quad \text{classification} = \text{'}\text{damaged}' \]

\[ \text{else:} \]

\[ \quad \text{classification} = \text{'}\text{healthy}' \]
### Sequence reconstruction

N = 100
total_error = 0
for window in sequence:
    X = []
    for i in range(1:N):
        X[i] = stochastic_predict(window)
    X_mean = mean(X)
    X_2std = 2*std(X)
    error = mean_squared_error(window, X_mean)
    for e in error:
        # cutoff
        if e > 0.33731913566589355:
            e = 0.33731913566589355
        # drop error within uncertainty
        if e < X_mean−X_2std or e > X_mean+X_2std:
            e = 0
    total_error += sum(error)

### Classification

if total_error > 42.65411979705095:
    classification = 'damaged'
else:
    classification = 'healthy
Chapter 4

Result

The main results of this thesis can be seen in table 4.2. Each row in the table represents a damage case. Cells marked with gray mark when over 5% of sequences are rejected and therefore the bridge is determined to be in a damage state, since we know that only 5% of healthy-state sequences were rejected. Since there are varying number of sequences available for each damage case, the results have to be interpreted carefully. This is detailed in the discussion chapter. Final parameters of the model are shown in table 4.1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input size</td>
<td>1458</td>
</tr>
<tr>
<td>Encode layer width</td>
<td>256</td>
</tr>
<tr>
<td>Z Code layer width</td>
<td>128</td>
</tr>
<tr>
<td>Decode layer width</td>
<td>256</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.1</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.0003</td>
</tr>
<tr>
<td>Weight initialization</td>
<td>Kaiming normal</td>
</tr>
</tbody>
</table>

The presented method in chapter 3 along with algorithms in chapter 3.4 is able to confidently detect damage cases 5-7, 9-11 and 13-14, where all three methods determine the state to be damaged. Note that the three different methods of detecting damage are all using exactly the same trained model.
Table 4.2: The percentage of rejected sequences of the different damage cases. Grayed cells mark where the bridge state is determined to be damaged, since over 5% of sequences are classified as damaged.

<table>
<thead>
<tr>
<th>Damage</th>
<th>Description</th>
<th>Uncertainty</th>
<th>Mean</th>
<th>Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Healthy state (100 sequences)</td>
<td>1.9%</td>
<td>1.0%</td>
<td>1.0%</td>
</tr>
<tr>
<td>1</td>
<td>Lowering of pier, 20mm</td>
<td>5.7%</td>
<td>14.3%</td>
<td>51.4%</td>
</tr>
<tr>
<td>2</td>
<td>Lowering of pier, 40mm</td>
<td>0.9%</td>
<td>6.4%</td>
<td>24.5%</td>
</tr>
<tr>
<td>3</td>
<td>Lowering of pier, 80mm</td>
<td>4.2%</td>
<td>4.2%</td>
<td>29.2%</td>
</tr>
<tr>
<td>4</td>
<td>Lowering of pier, 95mm</td>
<td>5.0%</td>
<td>5.0%</td>
<td>45.0%</td>
</tr>
<tr>
<td>5</td>
<td>Lifting of pier, tilt of foundation</td>
<td>13.6%</td>
<td>18.2%</td>
<td>50.0%</td>
</tr>
<tr>
<td>6</td>
<td>Spalling of concrete at soffit, 12 m²</td>
<td>17.4%</td>
<td>17.4%</td>
<td>30.4%</td>
</tr>
<tr>
<td>7</td>
<td>Spalling of concrete at soffit, 24 m²</td>
<td>20.0%</td>
<td>25.0%</td>
<td>35.0%</td>
</tr>
<tr>
<td>8</td>
<td>Landslide of 1m at abutment</td>
<td>1.1%</td>
<td>3.4%</td>
<td>6.8%</td>
</tr>
<tr>
<td>9</td>
<td>Failure of concrete hinge</td>
<td>6.8%</td>
<td>9.1%</td>
<td>15.9%</td>
</tr>
<tr>
<td>10</td>
<td>Failure of 2 anchor heads</td>
<td>20.8%</td>
<td>29.2%</td>
<td>37.5%</td>
</tr>
<tr>
<td>11</td>
<td>Failure of 4 anchor heads</td>
<td>8.2%</td>
<td>11.0%</td>
<td>12.3%</td>
</tr>
<tr>
<td>12</td>
<td>Rupture of 2/16 tendons</td>
<td>0.0%</td>
<td>0.0%</td>
<td>6.7%</td>
</tr>
<tr>
<td>13</td>
<td>Rupture of 4/16 tendons</td>
<td>10.0%</td>
<td>10.0%</td>
<td>10.0%</td>
</tr>
<tr>
<td>14</td>
<td>Rupture of 6/16 tendons</td>
<td>11.1%</td>
<td>14.8%</td>
<td>22.2%</td>
</tr>
</tbody>
</table>
Figure 3.11 shows the distribution of reconstruction error of the healthy-state unseen testset. This is the crucial information used to compare an unknown state to. The 95% error threshold is determined from this as described in chapter 3. Figure 4.1, 4.2 and 4.3 show the complete data from the damage case detection described in table 4.2. Please note that each damage case has different number of sequences, since the recorded time period varied for each damage case. For example, damage case 2 had over 100 recorded sequences, and damage case 13 had only 10.

Figure 4.1: All damage-case sequence classifications visualized for the uncertainty method.
Figure 4.2: All damage-case sequence classifications visualized for the mean method.
Figure 4.3: All damage-case sequence classifications visualized for the standard method.
Chapter 5

Discussion

The result presented is very promising for real life usage in a structural health monitoring system of a bridge. The method of using the same model in several ways is a novel approach to structural health monitoring, utilizing the trained model to a much fuller extent. The standard method is more performant in detecting damage, while the uncertainty method is much more cautious in the damage assessment. When the uncertainty method determines the bridge to be in a damaged state, it is a very robust determination. This is hugely important in real life usage of a structural health monitoring system. Realistic damage cases on the full-data Z-24 benchmark were detected.

A very fascinating result is that being better at reconstruction does not mean being better at detecting damage. The mean method achieves lower reconstruction error in all cases than the standard method. This does not mean that the mean method is better at detecting damage, in fact it is the opposite that is true. The standard method reconstructs healthy sequences adequately, but reconstructs damaged sequences poorly, and is therefore better at detecting the difference. This does not mean that the standard method overfitted on the training data, since it does not achieve remarkably lower reconstruction error on the training dataset compared to the testing and validation datasets. The mean model and uncertainty model generalize better to the damaged sequences, and therefore achieve worse damage detection.
5.1 Detected damage cases

Why is the method failing for certain damage cases? As seen in the results table 4.2, the damage detection did not confidently reject cases 1-4, 8 and 12. Note that the standard method rejected all damage cases, but the uncertainty was too great for the uncertainty method to reject these cases.

The lowering of the pier is a rare occurrence in a real life situation, but not unheard of. The severity of such an event depends a lot on the specific design of the structure. In a 3 span bridge with reasonable stiffness and length, as the Z-24 bridge, a 40 mm lowering will not induce a significant damaging force. The force would be roughly equivalent to a heavy truck passing. The lowering of a pier could still be a dangerous scenario, especially since it will weaken the bridge in an opposite direction of where the bridge is design to carry load. This will of course be heavily dependent on the structural design of the bridge. The damage detection does not confidently reject this scenario.

It is interesting to note that the damage detection confidently rejects cases 6-7, spalling of concrete at soffit. Such events are very common, and are not immediately dangerous as they do not contribute to bearing capacity of the concrete. It is possible that the sensors were located closely to the affected area, and therefore the damage detection more sensitive to the changes.

The method does not confidently reject case 8, landslide of 1 m at abutment. Such an event will have a specific effect on the bridge response in particular cases, since the landfill supports the bridge only in a certain load scenario. During the damage cases, the bridge was closed to passing traffic. This could explain that the damage was not detected, since the agitation of the bridge was only coming from traffic passing under the bridge which not as substantial.

Damage case 12, rupture of 2/16 tendons, was not rejected confidently. The rupture of tendons is rare and possibly dangerous but is very hard to detect from standard visual inspection. Therefore it might be more common than we think. The main reason that this damage scenario was not detected is that there were very few sequences, only 10 - corresponding to only 120 minutes. It is therefore reasonable to say that the damage detection could have been more accurate given a greater number of sequences.
5.2 Damage sensitivity

The presented method faces a well known challenge in the area of vibration based structural health monitoring, that damage is best assessed when the bridge is actively vibrating. This is clearly seen in figure 5.1 where the reconstruction error drops during the night when the bridge is in low use.

![Figure 5.1: The reconstruction error of a healthy-state data. Here it is clearly shown that the reconstruction error is dependent on activity, as the error drops notably down overnight.](image)

A big strength of this method is the resistance to false positives. As described in section 2.1, a well known challenge in structural health monitoring is the limiting factor of the training set capturing all possible cases. Therefore one of the biggest challenges of a structural health monitoring system is false positives. The Z-24 benchmark famously contains data when the temperature drops below freezing, which has a big effect on the behaviour of the bridge (Bart Peeters and Guido De Roeck 2001). It is a challenge to assess environmental effects versus damage events, as bridges can present drastic seasonal changes (Gonzalez and Karoumi 2014). In this method, such events as freezing temperatures receive no special treatment. These events are included in the training and test datasets, and therefore included in the 95% rejection threshold. The model is therefore able to assess such events, and better assess environmental changes versus damage cases. See figure 5.2 to compare a healthy state reconstruction to a damage state reconstruction.

Another matter to keep in mind when interpreting the results presented in table 4.2 is to note that some of the damage cases only contained a few data samples. To be able to successfully detect a damage rejection over 5% we would need more than a handful samples. For example, damage case 2 contains over 100 sequences, but damage case 13 contains only 10. The detected damage cases in shown in table 4.2.
are therefore dependent on this issue with the damaged-state dataset.

A final note, is that the healthy-state training data was captured while the bridge was in full use. Cars could pass over the bridge, and under. When the bridge was damaged, it was closed for traffic. Therefore, for every damage case the bridge was only excited by underneath traffic which results in much smaller vibration. Therefore the damage detection is even more challenging, and requires a more sensitive damage detection.

![Figure 5.2](image.png)

**Figure 5.2:** All three model methods compared for a given input. The plotted data only shows one sensor and a 50-millisecond window for clarity. Damage case 7 below, healthy state above. Take note of the increasing uncertainty in the damage scenario.
5.3 Ethical implications

This thesis presents an end-to-end method of detecting damage in a real life bridge using only sensor measurements. Even though the results are a promising step into a more modern structural health monitoring of bridges, I believe that we are far from being able to replace standard monitoring methods of bridges. Using the presented method as a replacement for standard bridge inspection would be reckless and possibly life-threatening. What this thesis presents is a method that could complement standard procedure, and possibly improve the speed and accuracy of damage detection. If the method confidently predicts a damaged state, the prediction would have to be confirmed manually.

Damaged bridges are not a thing of the past. On August 1, 2007 the I-35W bridge in Minneapolis collapsed, killing 13 and injuring 145. On 14 August 2018, a major bridge in Genoa, Italy partially collapsed, killing 43 people. One can wonder if a modern, data based structural health monitoring system alongside strict maintenance monitoring could have prevented these events.
Chapter 6

Conclusion

The results are very promising for the field of bridge structural health monitoring. A modern neural network model trained on the complete sensor data was able to detect several realistic damage cases on the Z-24 bridge benchmark (Reynders and Guido De Roeck [2014]), which has been used in research for over 20 years. Due to the size of the full data and computational limitations, few researchers have presented methods of structural health monitoring on the full data. To our knowledge, this is the first time a model has been trained on the full healthy-state data and successfully detect damage cases while taking no special care of freezing temperature variations of the healthy-state data.

The method is performant, and could very reasonably be used in an operational bridge for damage detection and is a prominent step towards modern infrastructure. The methods ability to detect damage in realistic damage scenarios of a real bridge has many economic, social and ethical aspects. Such a system could possibly detect damage before a traditional bridge inspector could, and prevent a costly repair.

Society depends on infrastructure to be safe, and it it quite possible to imagine that this method could save lives if no other system is at hand. Reduction in maintenance and operational cost or the extension of structural lifetime could potentially save enormous amounts of government spending on infrastructure.

6.1 Case for real life use

The method has potential for real life usage. The model can reconstruct a full 10 minute sequence and classify it in the order of seconds, on a
standard modern personal computer. This means that is computation-
ally very fast, and can be used in real time operation. The training of the
model on the other hand requires considerable computing power. The
model for this thesis was trained on a cloud computer with 16 CPUs,
50GB memory and a 16GB GPU NVIDIA Tesla P100. The training data
was around 50 GB.

The time scale of damage detection is more nuanced, since the
damage detection is based on evaluating multiple sequences. The
damage sensitivity is also dependent on bridge activity. In the case of
considerable damage, the model would start rejecting the 10 minute
sequences in succession. For the model to assess damage robustly, it
would need multiple 10 minute sequences. Therefore the time scale of
reliable damage detection is in the order of hours, possibly days.
Bibliography


In: Medical image analysis 36, pp. 61–78.


