Monitoring Flexural Fatigue Life of Asphalt Concrete by Piezoelectric Sensors

(Strategy for fatigue life prediction and damage detection based on load history on any asphalt matrix)

Syed Baqir Ali Kazmi

TU Delft

Monitoring Flexural Fatigue Life of Asphalt Concrete by Piezoelectric Sensors

(Strategy for fatigue life prediction and damage detection based on load history on any asphalt matrix)

by

Syed Baqir Ali Kazmi

Student Name	Student Number
Syed Baqir Ali Kazmi	4436474

Thesis Committee:

Dr. Mohammad Fotouhi (Chair)
Dr. Xueyan Liu
Dr. Yi Li
MSc Ali Ghaderiaram

Company Supervisor: Robbert Naus
Project Duration: March, 2023 - October, 2023
Faculty: Faculty of Civil Engineering, Delft

Cover picture [1]

TU Delft

Preface

This thesis report represents the culmination of my graduation project conducted at TU Delft. My collaboration with Dura Vermeer, facilitated by the university, provided invaluable support in the form of material resources and laboratory data. The selection of the research topic was driven by my profound fascination with Machine Learning and Artificial Intelligence, cutting-edge technologies rapidly reshaping various industries. Consequently, this thesis serves as a methodological framework for predicting the fatigue life of asphalt specimens with the ultimate objective of full automation for monitoring purposes in the future. It is worth noting that this research topic was not officially proposed, and as such, I assumed full responsibility for defining the research problem and executing the project under the gracious guidance of Dr. Muhammad Fotouhi.

This thesis lays the groundwork for further exploration in the domain of structural health monitoring. It elucidates the methodology employed for harnessing machine learning in predictive applications. However, it is essential to underscore that the results derived from the asphalt experiments necessitate further verification through extensive laboratory testing. The research herein is based on a limited number of beams subjected to testing in a pneumatic 4PB machine, which is inherently less precise than a hydraulic 4PB machine. Moreover, different asphalt configurations demand comprehensive examination which may deviate from Teflon tests since Machine Learning models are trained using Teflon data. However, this is not seen for the majority of particular beams that are used.

I would like to begin by expressing my gratitude to the divine God for guiding me throughout this journey and in the future. In addition, I extend my heartfelt appreciation to the members of the thesis committee for their invaluable contributions to the realization of this project. I reserve special recognition for Ali Ghaderiaram and Dr. Muhammad Fotouhi, who not only served as mentors but also as friends, displaying unwavering patience and offering their invaluable assistance, both of which were instrumental in bringing this thesis to fruition. I would also like to acknowledge my friends for their companionship during breaks, as well as my parents and family for their unwavering trust, support, and guidance.

I sincerely hope that you find this thesis to be an engaging and informative read.

Syed Baqir Ali Kazmi
Delft, November 2023

Summary

The primary objective of this dissertation is to assess the viability of developing an effective and novel supervised Structural Health Monitoring (SHM) methodology for the detection of damage and the prediction of fatigue life in materials such as asphalt concrete, contingent upon knowledge of their S-N or Wöhler curves. Detecting damage and determining when maintenance should be applied to asphalt pavement using sensor data poses significant challenges, necessitating robust and efficient methodology. The success of such a methodology could result in substantial savings by enabling timely inspections and interventions, optimizing resource utilization, including labor allocation, where it is most needed.

The approach begins with a comprehensive literature review to establish the requisite background knowledge and state-of-the-art concepts. Subsequently, a testing program is formulated, employing piezoelectric sensors, specifically Lead Zirconate Titanate (PZT) and Polyvinylidene Fluoride (PVDF), in a four-point bending (4PB) constant strain test. PVDF (Polyvinylidene fluoride) is not elaborated upon in this study, except in Appendix E, due to the suboptimal voltage generated by the PVDF sensor. This four-point bending testing setup and procedure are identified as the most suitable for estimating fatigue parameters after a thorough review of fatigue testing methods. The methodology further incorporates supervised machine learning, as machine learning techniques excel in addressing regression problems, encompassing the estimation and prediction of parameters. Machine learning simplifies the complexity of the problem by training models based on provided data, with training data being the primary limiting factor. To facilitate this, Teflon beams with a stiffness similar to asphalt beams were employed to gather extensive data for training machine learning models by doing 4PB tests with different strain levels and frequencies.

Converting sensor data into strain, especially in bending mode, requires a sound understanding of mechanical, electrical, and structural engineering. However, machine learning offers the advantage of utilizing sensor data to train ML models for accurate strain prediction. Machine learning also possesses the capability to adapt to multiple S-N curves under varying conditions, such as temperature, frequency, and humidity. This adaptability is particularly advantageous, as one of the future objectives of this study is to employ piezoelectric sensors in in-service pavement.

As demonstrated in previous research [2, 3, 4, 5], higher strain levels lead to shorter service life for asphalt. The combination of different cyclic loads and frequencies induces varying flexural tensile strain in asphalt. Consequently, knowledge of strain levels at each load cycle is imperative for the accurate prediction of asphalt fatigue life. Moreover, as asphalt exhibits viscoelastic properties, its behavior is influenced by temperature and moisture conditions [6, 7, 8]. To provide a reliable estimate of strain that accounts for all these parameters within the constraints of this thesis time frame, a machine learning approach is chosen.

The methodology is tested on asphalt beams following the successful training of machine learning models. These models are employed to predict strain based on the voltage amplitude and frequency from the PZT sensor at various strain levels. The successfully predicted strain value per load cycle is then used to plot damage curves and predict fatigue life using established fatigue life models with the Cheng model giving the best result. The results of various machine learning models are compared, with Extra Trees Regression, Bagging Regression, Neural Network, GradientBoosting Regression, and Catboost Regression identified as the most suitable models for predicting strain based on sensor data. From the comparison of fatigue life models, the Cheng model is identified as the most appropriate for use within this methodology, while either Miner's rule or the Bodin model can be employed to derive damage curves. In addition to the main goals of this research, COMSOL numerical analysis was performed with results confirming a similar

pattern as obtained by the Teflon beam 4PB test. Furthermore, when integral voltage amplitude was plotted against maximum strain (figure D.4), it was found that the curve stayed the same until a frequency of 5[Hz] confirming earlier research [9].

Contents

List of Figures

List of Tables

Nomenclature

If a nomenclature is required, a simple template can be found below for convenience. Feel free to use, adapt or completely remove.

Abbreviations

1D	One-Dimensional
2D	Two-Dimensional
Abbreviation	Definition
AC	Asphalt Concrete
ANN	Artificial Neural Networks
2PB	Two-Point Bending
3PB	Three-Point Bending
4PB	Four-Point Bending
CNN	Convolution Neural Networks
COMSOL	Finite Element Package
DL	Deep Learning
ITFT	Indirect Tensile Fatigue Test
IT	Indirect Tension
FEA	Finite Element Analysis
FPBT	Four Point Bending Test
ML	Machine Learning
MLP	Multi Layer Perceptron
OLS	Ordinary Least Squares
PI	Physik Instrumente
PFG	Piezo-Floating-Gate
PVDF	Polyvinylideenfluoride
PZT	Lead Zirconate Titanate
RNN	Recurrent Neural Networks
S-N	Strain amplitude-Life
SHM	Structural Health Monitoring
UT	Uniaxial Tension
UC	Uniaxial Compression
WSNs	Wireless Sensors Networks

Symbols

Symbol	Definition	Unit
N_f	Number of cycles until failure	[-]
ϵ	Strain	[-]
ϵ_t	Tensile Strain	[-]
T	Temperature	[°Celcius]
RP	Rest Period	[seconds]
E_0	Material stiffness	[MPa]
D	Damage	[-]

1

Introduction

1.0.1. Background and motivation

There is an increase in the trend for the maintenance budget of roads which reached 1.2 billion euros in 2011 even more than the previous year [10]. Due to the fatigue of structures, it is expected that more maintenance will be required in the future. Unnecessary or improper monitoring can even make this worse resulting in wasting money or using resources in places where it is not required. Another problem is more labor work and less money. An efficient way of working could be working only where it is necessary and in order to know where work is needed, we need smart and automated way of detection instead of general periodical inspections.

There are three types of maintenance operations [11]:

- Preventive Maintenance
- Corrective Maintenance
- Emergency Maintenance

This concept is elaborated upon in the figure 3.1 below, presenting the required maintenance interventions corresponding to the evolving pavement condition over time or in relation to the cumulative number of load cycles imposed by traffic. It is worth noting that current maintenance and damage detection techniques predominantly encompass manual surveys or automated surveys utilizing specialized equipment, often requiring skilled personnel for the execution of these tasks.

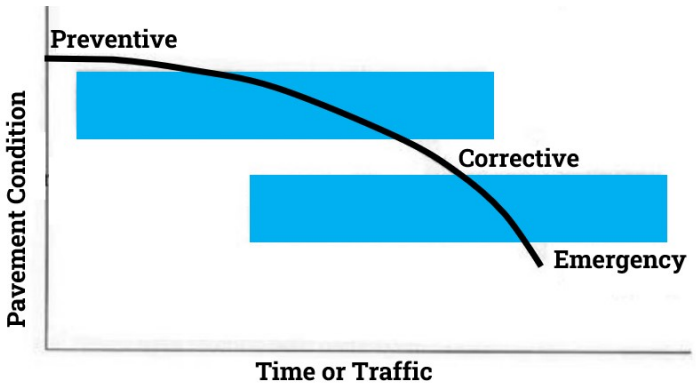


Figure 1.1: Maintenance categories reproduced from [11]

It is therefore vital to have good prediction models not only for rutting damage but also for fa-

tigue. In the Netherlands, different parties do the maintenance of different types of roads such as Rijkswaterstaat for the national roads, provinces for the provincial roads, and municipalities for the local roads [12]. This motivates ideas to be invented such as using sensor technology and machine learning to predict the fatigue life of asphalt which is not only innovative but has great potential in coping with the challenges mentioned earlier. Furthermore, the concepts of different maintenance operations are discussed below.

Preventive Maintenance is an essential approach to extending pavement lifespan through planned surface treatments, addressing wear, and delaying failures. It aims to minimize routine maintenance needs and control damage caused by environmental factors like oxidation and surface cracking. Treatment frequency should be time-based, as environmental conditions remain stable. By adopting preventive maintenance, infrastructure longevity, and performance improvement, reducing the necessity for extensive repairs [11].

Corrective Maintenance responds to pavement deficiencies, such as friction loss, rutting, and cracking. It differs from preventive maintenance in cost and timing, as it addresses repairs when necessary and is more expensive. Being highly reactive, delaying corrective maintenance escalates pavement defects and costs. This increases the pavement's life cycle expenses, and activities involve structural overlays, milling, pothole repair, and crack repair [11].

Emergency Maintenance addresses urgent situations like blowouts or severe potholes that need immediate repair, ensuring safety and maintaining traffic flow. Temporary treatments stabilize the surface until permanent repairs can be implemented. In emergencies, safety and time take precedence over cost, making materials suitable for short-term solutions even if they are not ideal for preventive or corrective maintenance [11].

1.0.2. Problem statement

The present research aims to develop more robust and efficient automated methodologies for predicting the fatigue life of pavement structures. To this end, it is imperative to integrate advanced machine learning and artificial intelligence techniques with sensor technology, enabling a more precise and comprehensive analysis of pavement damage caused by vehicular traffic loads.

To progress toward the practical implementation of these innovative techniques in real-world pavement structures, it is imperative to execute experimental tests aimed at establishing the correlation between the sensor-generated data and actual pavement damage. In this research, a substantial focus is directed toward the development of such methodologies, alongside the integration of Machine Learning (ML) models, as well as the incorporation of pre-existing phenomenological models for fatigue prediction and damage assessment, with the overarching goal of predicting asphalt fatigue life and constructing damage curves.

Consequently, a practical and efficacious strategy involves the utilization of the SN (S-N) curve method. This method entails the execution of four-point bending fatigue tests and harnessing sensor data to predict the fatigue life of asphalt specimens. This predictive process is realized through the integration of sensor data, ML models, SN curve, cumulative damage models, and fatigue life prediction models. This pioneering approach not only streamlines the prediction of asphalt fatigue life but also factors in the influence of vehicular traffic loads, offering damage assessments on a per-load-cycle basis. The number of load cycles is quantified via strain measurements facilitated by sensors, derived from piezoelectric voltage, and validated against data acquired from LVDT during experiments. Since constant strain tests are employed, these measurements are further cross-referenced with predefined flexural strain values. The utilization of constant strain tests within pavement laboratory settings streamlines this process, obviating the need for cycle counting, as the predetermined number of cycles and frequencies utilized until failure simplifies data interpretation. However, it is advisable in real-world applications to im-

plement cycle counting algorithms such as the Rainflow cycle counting algorithm. These algorithms facilitate the determination of the number of cycles per voltage amplitude, where higher voltage amplitude signifies greater vehicular load and subsequently more strain, as well as the precise frequency. Once voltage and frequency are accurately determined, machine-learning techniques can be employed to convert this data into strain measurements. With this strain-per-cycle information, it becomes feasible to predict fatigue life and construct damage curves, thereby enhancing the predictive capacity of the methodology.

To summarize how this issue is addressed, the research involves performing a series of standard four-point bending fatigue tests according to **NEN-EN 12697-24** to generate the SN curve and the master curve, as well as testing a few pavement specimens with the sensor to acquire sensor data in terms of voltage by changing strain levels and frequency. Subsequently, algorithms are developed to transform the sensor data (combination of voltage and frequency) into strain data. Afterward, simple phenomenological models and damage models are used to predict the fatigue life based on the SN curve, damage models, and fatigue life prediction models. These damage models are based on past literature and are critical in enabling more accurate and reliable predictions of pavement fatigue life.

1.0.3. Research questions

1.0.3.1. Main Question

How to develop a methodology using supervised learning techniques that can effectively detect early-stage damages in material using piezoelectric sensors?

From the main question, sub-questions arise which will be explored in this research and answered via literature study, methodology, results, and conclusions. Some of these sub-questions are very basic and require and instructs to study for example material property to more advanced questions like understanding fatigue crack and predicting fatigue life.

1.0.3.2. Sub Question

- What
    1. What is the mechanism of fatigue in asphalt?
    2. What are the fatigue life characteristics of asphalt?
    3. What is the piezoelectric sensor and how does it function?
    4. What are the techniques used for fatigue life prediction in asphalt?
    5. What are the essential requirements for predicting the fatigue life of an asphalt specimen?
- How
    1. How can the fatigue of asphalt be detected?
    2. How can piezoelectric sensors be utilized to predict the fatigue of asphalt?
- Why
    1. Why is early detection of fatigue damage important in asphalt?
    2. Why are piezoelectric sensors used for fatigue life prediction and detection in asphalt?
    3. Why is there a need for developing a new method for fatigue life prediction and detection in asphalt?
    4. Why machine learning is required for this study?
- Who
    1. Who are the potential beneficiaries of this technology in the field of asphalt?
- Where

1. Where is this technology expected to be most effective in the field of asphalt?
2. Where is the relevance and need for asphalt fatigue testing?

1.0.4. Research scope and objectives

This research will center its attention on crafting a resilient and automated framework for the prediction and assessment of the flexural fatigue endurance of asphalt pavements. This framework leverages advanced piezoelectric sensor technology, machine learning algorithms, and cumulative fatigue damage models. It is important to note that the scope of this study will be confined to laboratory experiments, aimed at establishing the groundwork for future investigations. These subsequent studies may involve the incorporation of more sophisticated models or the development of novel methods that directly utilize piezoelectric voltage or the resulting strain measurements as done in this research for real-world field applications.

The primary goal of this research is to establish a comprehensive framework capable of forecasting fatigue life and monitoring fatigue damage through the utilization of data acquired from piezoelectric sensors. The proposed methodology entails the transformation of sensor data, initially in the form of voltage amplitude and frequency, into strain measurements through the application of Machine Learning techniques. Subsequently, these flexural tensile strain values will be employed to compute and visualize damage curves and aid in making predictions about the asphalt's fatigue life based on the strain data. It is noteworthy that this general framework's applicability extends beyond asphalt and includes other cementitious mixes such as Engineered Cementitious Composites (ECC) and concrete provided that their SN curve is known. Moreover, the framework is not restricted to the realm of traffic loading but possesses versatility for various applications and contexts.

The vision for this research is to enable real-time structural health monitoring of asphalt pavements and other cementitious mixes by providing a comprehensive framework for fatigue life prediction and damage monitoring. The developed framework will provide real-time data from the fatigue sensor and a damage accumulation curve that updates after each cycle, allowing for the prediction of structural life as the number of cycles changes. This will be achieved by developing a robust and automated approach for predicting fatigue life and monitoring fatigue damage in asphalt pavements and other cementitious mixes.

The specific objectives of this research are to gain a better understanding of fatigue in asphalt, to evaluate the benefits of piezoelectric sensors for fatigue monitoring, to develop a framework to convert sensor data to strain, to use existing cumulative damage models after a comprehensive literature study for plotting damage curves, to translate the strain curve to a damage curve using damage models and to predict remaining life based on the strain curve, damage curve, and possibly using a master curve or SN curve. The successful completion of these objectives will lead to a more robust and automated approach for predicting fatigue life and monitoring fatigue damage in asphalt pavements and other cementitious mixes using a piezoelectric sensor by just attaching it to the substrate. Furthermore, the developed framework will be extendable to other types of loading spectra, including but not limited to, environmental loading, wind loading, and seismic loading. The current framework is however limited due to limitations of time and resources, however, it nevertheless lays a good foundation for future study.

1.0.5. Research approach

The present study employs a methodology and research approach that is focused on developing a robust and automated framework to predict and evaluate the flexural fatigue endurance of asphalt pavements. The objective of this research is to overcome the challenges of traditional approaches to fatigue testing and monitoring in asphalt pavements that are often time-consuming, costly, and prone to errors. The proposed methodology utilizes piezoelectric sensor technology, machine learning algorithms, and cumulative fatigue damage models to provide an efficient, precise, and dependable solution for evaluating the fatigue resistance of asphalt pavements.

To establish the foundation for future research, two batches of asphalt prismatic beams are prepared and tested in the laboratory. The first batch is subjected to four-point bending according to NEN-EN 12697-24 to obtain an SN curve. This curve is a standard approach for characterizing the fatigue behavior of asphalt and provides valuable information about the fatigue performance of asphalt pavements under different loading conditions, such as the maximum stress level, the number of loading cycles to failure, and the fatigue life. The second batch is tested under sinusoidal and constant strain loading using sensors to validate the developed methodology. Sensor generates voltage as an output which is measured using an oscilloscope which can be post-processed using Python to extract voltage amplitude also known as peak to peak voltage and store them as values in the list with other voltage amplitudes of same frequency that they are taken from. Hence, each value in the list represent voltage amplitude based on different strain levels for same frequency. Different strain levels are tested in the lab which gives different voltages based on the given strain levels for the same frequency. The resulting data, after converting the sensor data to strain using supervised machine learning algorithms, will be utilized to predict the fatigue life of asphalt beams using SN curve and cumulative fatigue/damage models.

This research employs a variety of supervised machine-learning algorithms to facilitate training with the goal of estimating or predicting strain based on a multitude of voltage and frequency combinations. To assess and distinguish the performance of these algorithms, a systematic comparative analysis is conducted, utilizing the R-squared method. This method, widely recognized in the realm of regression analysis, provides a quantitative measure of the model's goodness of fit concerning the observed data. R-squared values range between 0 and 1, with higher values indicative of a stronger and more precise fit between the model and the actual data. Also other methods are used in order assess quality of the ML algorithms such as Root Means Squared Error (RMSE), Mean Absolute Error (MAE), etc.

For the training phase, Teflon beams, possessing stiffness similar to asphalt beams, is employed, as it offers a safer alternative due to the potential risks associated with using asphalt beams, which have limited availability. Subsequently, during the testing phase, asphalt beams are employed. Once strain data is collected, it can be incorporated into phenomenological models that establish a connection between material S-N curves, damage, and fatigue life. These models also facilitate the construction of damage curves, with the objective of identifying the most suitable model among the alternatives.

The incorporation of existing models serves to demonstrate the potential of the proposed methodology, which can be further developed into more advanced and fully automated models. Supervised machine learning algorithms can also undergo further automation to transition into reinforced and unsupervised variants, requiring minimal user input or oversight. The successful completion of this study is positioned to enhance our deeper understanding of fatigue mechanisms in asphalt, evaluate the practicality and effectiveness of piezoelectric sensors for fatigue monitoring, and deliver a resilient and automated framework for predicting fatigue life and overseeing fatigue damage in asphalt pavements as well as other cementitious mixes. This advancement has the potential to greatly contribute to the durability, safety, and sustainability of such structures, particularly in demanding conditions characterized by heavy traffic and challenging environmental factors.

The proposed methodology carries the potential to significantly enhance the longevity, safety, and sustainability of asphalt pavements, fortifying their performance under the demands of heavy traffic and challenging environmental conditions.

1.0.6. Research outline

The forthcoming thesis shall be organized into a series of main chapters and sections, each encompassing numerous sub-sections. The preface will provide an all-encompassing summary of the study and its purpose. The summary section will furnish a concise overview of the findings, whilst the nomenclature segment will serve to define any technical terms or acronyms to be employed throughout the dissertation.

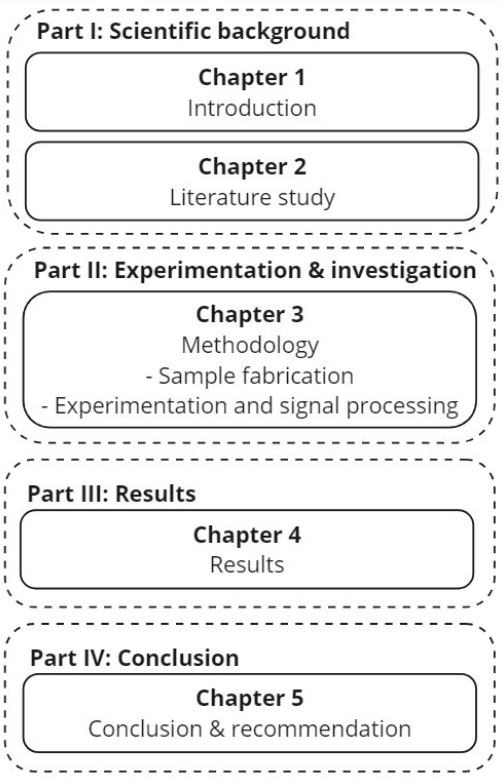


Figure 1.2: Thesis outline

Chapter 1, Introduction, shall be comprised of several sub-sections, including background and motivation, problem statement, research scope and objectives, research questions, research approach, and research outline. This chapter will serve to provide an initial exposition of the topic and create a foundation for the subsequent research.

Chapter 2, Literature Study, will survey the existing literature and research on several pertinent topics, including sensors, structural health monitoring, asphalt, fatigue, and machine learning. This chapter shall provide a contextual framework for the research and highlight how the current study augments existing research in the field.

Chapter 3, Methodology, will elucidate the experimental methods and procedures employed in the study. This chapter will be divided into several sub-sections, including the four-point bending test, piezoelectric sensor, cumulative damage models, fatigue life prediction, and cumulative damage and structural health monitoring. These sub-sections shall expound on the methodology adopted in the study and exhibit how the data was collected and analyzed.

Chapter 4, Results, shall present the findings of the study. This chapter will include several subsections outlining the outcomes of the experiments conducted in Chapter 3. The chapter will also incorporate tables, graphs, and other visual aids to help illustrate the results.

Chapter 5, the Conclusion, will concisely summarize the study's findings and engage in a comprehensive discussion of their implications. This chapter will also address the study's limitations and offer insights into potential avenues for future research, along with critical discussions. The References section will provide a comprehensive list of all sources cited in the thesis, ensuring proper attribution and credit for the scholarly work cited throughout the research.

In essence, the thesis will be partitioned into five chapters, with each chapter composed of various sub-sections. The Introduction chapter will establish an initial framework for the research, the Literature Study chapter will survey existing research in the field including reviewing all different fatigue life prediction models, the Methodology chapter will detail the experimental procedures utilized in the study, the Results chapter will present the findings of the study, and

the Conclusion chapter will encapsulate the findings and engage in an elaborate discussion of their implications. The thesis shall culminate with a catalog of references cited in the text.

2

Literature study

This chapter serves as a comprehensive review of various cutting-edge research papers, which collectively establish the foundation for this study. It delves into the requisite background knowledge essential for the development of a supervised methodology aimed at harnessing the potential of piezoelectric sensors in the realms of fatigue life prediction and damage curve visualization. The primary goal of this chapter is to explore different fatigue testing methods and various fatigue life models that enable the projection of the service life of asphalt concrete based on laboratory tests, including the calculation of damage per cycle. Special emphasis is placed on investigating phenomenological models.

In addition to these aspects, the concept of Structural Health Monitoring (SHM) is introduced, along with an examination of diverse sensor types. This chapter also delves into the underlying principles and concepts related to SHM, which have greatly contributed to the formulation of the novel methodology. Notably, it provides in-depth insights into piezoelectric sensors, specifically Lead Zirconate Titanate (PZT), and investigates the pertinent mathematical equations governing PZT behavior, shedding light on the generation of electric charge and its response to various mechanical modes, such as bending and compression.

Furthermore, to gain a comprehensive understanding of the role of machine learning (ML) in this context, the chapter explores different ML algorithms, elucidating the concepts of various ML techniques and the structure of an ML pipeline.

In Section 2.1, an extensive review of Structural Health Monitoring (SHM) is provided. Section 2.2 delves into sensors and their various types. Subsequently, Section 2.3 scrutinizes asphalt as a material, encompassing an exploration of different asphalt failures in Section 2.3.1. Following this, Section 2.4 delves into fatigue models, specifically focusing on asphalt fatigue failure in Section 2.4.2.1. The study then shifts to different test setups in Section 2.5, elucidating loading modes in Section 2.5.1, the effect of specimen size in Section 2.5.2, the impact of temperature in Section 2.5.3, the influence of loading pattern in Section 2.5.4, and the role of loading frequency in Section 2.5.5. Section 2.6 rounds off the literature review by delving into machine learning models and approaches for assessing the fatigue life of asphalt concrete.

2.1. Structural health monitoring

Structural health monitoring (SHM) is an emerging field in Civil engineering with the main purpose of detecting early damage. Pavement health monitoring comes also under the category of structural health monitoring. One of the most common sensors used in SHM are strain gauges [13] which is not very practical for regular in-service pavements as they require energy in order to read its data. Lead zirconate titanate (PZT) sensors provides an alternative to be used for pavement since they can be used as passive sensor where no energy is required as it itslef have ability to generate charge. They do not necessitate external power sources and can even function as

energy harvesters. Using PZT sensor generates data that can be used to detect deformations, flexural strain and damage of the layer where it is deployed.

Many methods have been developed for structural health monitoring in Asphalt Concrete (AC) such as using self-powered Piezo-Floating-Gate (PFG) wireless sensors [14]. AC beam is tested under three-point bending and the results from the sensor are used to analyze the bottom-up cracking in AC pavement. The change in charge at the floating gates was used as an indicator of damage. The model setup used for this purpose is in the figure 2.1. It has already been concluded later that four-point bending is very similar to the real in-service asphalt and hence using four-point bending instead of three-point bending could be more practical. Similar study but instead of using PFG wireless sensors, PZT sensors were used but again for verification of sensor data, 3PB test is used instead of the 4PB test [15].

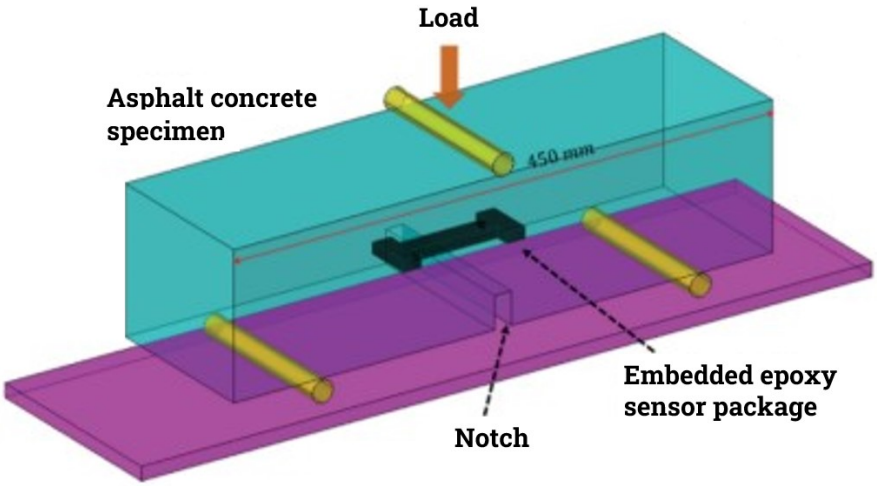


Figure 2.1: Schematic of the notched asphalt concrete specimen under three-point bending test [14]

Another method is embedding a piezoelectric transducer for reinforced concrete which uses structural mechanical impedance (ESMI) to detect any damage. In the impedance technique, the electromechanical impedance signal is measured by applying an electric voltage to PZT and measuring the corresponding output current [16] [17].

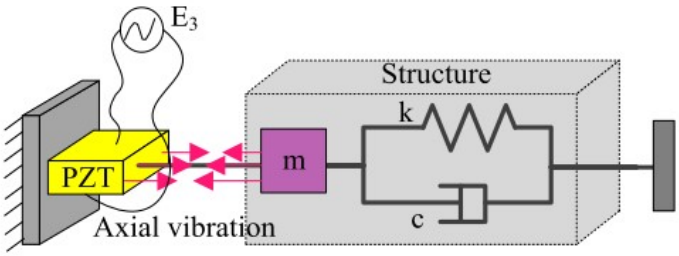


Figure 2.2: Typical one-dimensional electromechanical interaction model [17]

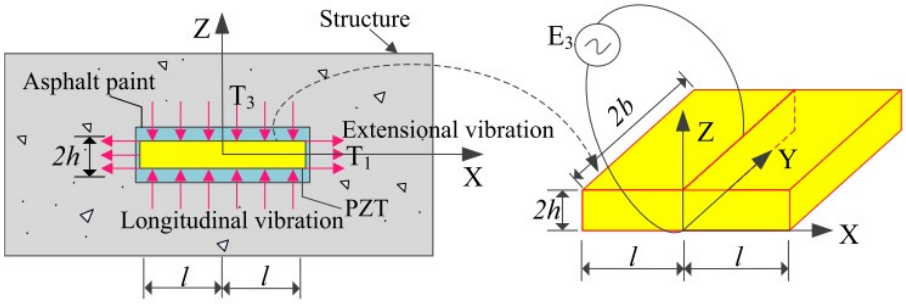


Figure 2.3: Schematic diagram of PZT embedded [17]

As seen from the figures 2.1, 2.2 and 2.3, this method deals either with 1D interaction or 2D embedded. It does not include a 4PB test setup or bonding sensor on the substrate and investigating its voltage neither the first source [14] does that. Another paper integrates the finite element method and probabilistic neural network (PNN) based on Bayesian decision theory for damage detection using self-powered wireless sensor [18]. This paper also inspires this research for using machine learning (ML) models as also described in the chapter methodology of this thesis. What it lacks is that it uses three-point bending and then uses a bridge gusset plate as a case study.

One other paper has used Fiber-Bragg-Grating (FBG) optical sensors for structural health monitoring of road pavement [19, 20]. This thesis is also inspired by this research as in this paper they have embedded it in the pavement to get real-time traffic-induced strain readings and at the same time monitor the road's structural health. Lots of the papers use the electrical impedance method to detect cracks or damage for real-time health monitoring method [21, 22, 23, 17, 24]. However, they do not use peak-to-peak voltage or do not give a general method in order to be able to use this method which is taken into consideration in this research.

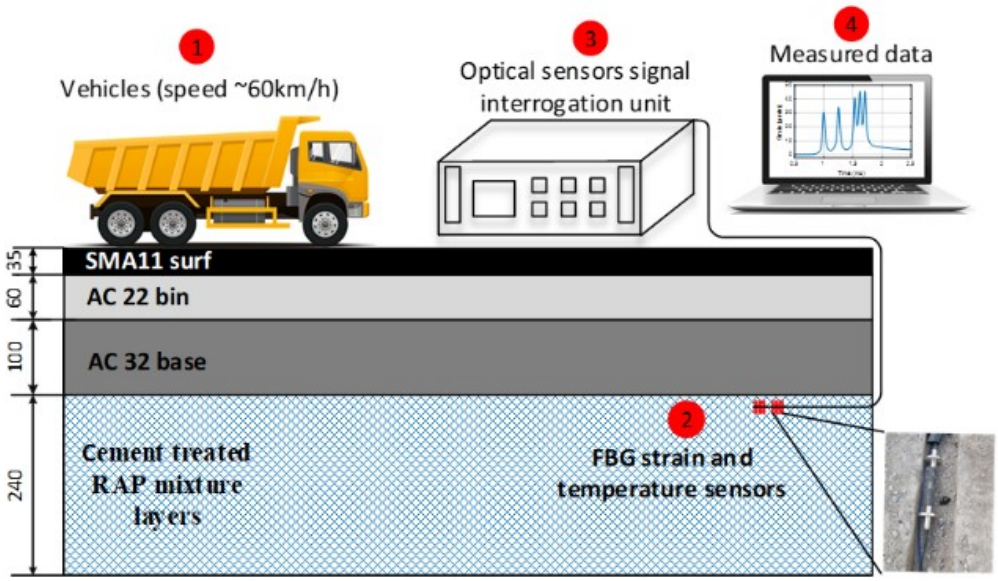
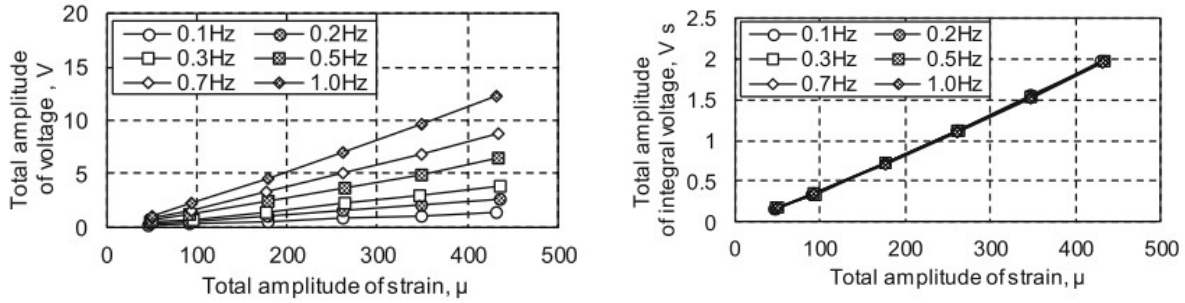


Figure 2.4: Schematic setup of using FBG optical sensors in a real field study [19] [17]

One of the papers that comes close to this research in terms of sensor output result is done by a few researchers where they used piezoelectric ceramic sensors on the bridge of steel material [9]. They then plotted the total value of integral voltage vs total amplitude of strain strain where they found one single line for all the frequencies that they used as shown in figure 2.5b. Similar

results are also found in this thesis however with the conclusion that that is only possible for small frequency bandwidth. Also, they plotted the total amplitude of voltage vs strain which is quite similar to what is found in this thesis.



(a) Voltage amplitude vs total strain amplitude plot

(b) Integral voltage [Vs] vs time [s] plot

Figure 2.5: Results of sensor data [9]

Piezoelectric material is also known to be sensitive to temperature [14]. It has been proved that both temperature and frequency have an effective impact on piezoelectric sensor data especially if it is used to detect low damage levels [25]. In the paper where they used FBG optical sensors, it has been found that in winter time (1 - 1.5 ° Celcius) the average relative strain is 2.5 times higher than in summer time (24.8 - 25.1 ° Celcius) at the recycled layer as shown in figure 2.4 [19].

Lots of the models either use 1D elements to embed the sensor in the material or use the impedance method for damage detection. This research takes another approach and makes use of peak-to-peak voltage also known as voltage amplitude received from the PZT sensor for a wide range of frequencies and uses ML models to produce strain based on the peak-to-peak voltage and frequency and then uses standard phenomenological asphalt fatigue models to predict fatigue life and damage of the substrate. This approach is what makes this research novel and different then the traditional method seen so far. Next, we see different types of sensors that are used in SHM of concrete-type structures [26].

2.2. Sensor

Sensors are used for SHM purposes as they play a vital role in gathering data for SHM purposes in detecting and measuring different quantities. The table 2.1 summarizes some of the most used sensor types [26].

According to the information presented in Table 2.1, the Piezoelectric (PZT) sensor stands out as the most appropriate type of sensor for potential use as a passive sensor on pavements requiring minimal equipment. Although this aligns with one of the overarching goals, it's essential to note that this aspect falls beyond the current research scope. Consequently, it has been determined that a piezoelectric PZT sensor will be utilized in this primary research phase to establish the necessary protocols and methodology that will enable the sensor to serve in this intended capacity.

The way it is used is also cheap as it does not require expensive equipment if it is used as a passive sensor since data can be transferred via Bluetooth. In this research however oscilloscope has been used but for the future goal, it will be used by itself in some type of proper network. Another interesting type of sensor is PVDF. It is also used in this research however it did not give good results. This is further discussed in chapter "Results" and PZT is more focused in subsequent sections. In this research, soft PZT is used in room temperature and since the tensile strain imposed by the machine is not so high, there are no risks in terms of sensor performance. Soft PZT exhibits greater flexibility compared to standard PZT, making it well-suited for this research where the substrate beam experiences significant vertical deflection based on given flexural tensile strain, due to a four-point bending machine.

Table 2.1: Sensor Types, Advantages, and Limitations

Sensor type	Input	Output	Advantages	Limitations
Piezoelectric	Pressure	Electric charge	Sensitive, wide frequency range, inexpensive, different sizes, can be used as a passive sensor	Fragile and brittle failure if tensile strength is exceeded, limited curvature dependent on the application, sensitive to vibration or acceleration, temperature-dependent properties
Temperature sensor	Heat	Surface charges	Wide temperature range, compact sizes	Fragile, environmentally affected sensor, sensor drift over time
Optical fibers	Opto-electronic signals	Displacements, strains, temperature	High bandwidth, low attenuation, and immune to electromagnetic interference	Sensitive to fiber leads causing small decay to the loss of entire data, expensive, difficult to be used as a passive and as an energy-harvesting sensor
Strain gauges	Electrical voltage	Resistance	Inexpensive, easy to install	Depends on aggregate size for asphalt beam for correct measurement, limited measurement range, calibration is required, and prone to noise
Magneto-strictive sensor	Electromagnetic energy	Mechanical energy	High flexibility, suitable for any climate range including harsh environments, high-frequency vibration	Limited resolution, magnetic field interference, installation complexity, and expensive
Accelerometer	Mechanical energy	Electrical energy	Very sensitive and high accuracy in damage detection	Prone to noise, very expensive, and even vandalism issues

In this section, the piezoelectric sensor is studied and state-of-the-art results are presented related to the piezoelectric sensor and its mathematical equations which describe the sensor and form the basis for understanding its fundamental behavior. There are different types of transducers such as magnetic, optical, electrochemical, mass-based and thermal transducers [27]. Sensor is a type of transducer but transducer are not necessarily sensor. Piezoelectric comes under mass-based transducer [27]. A transducer focuses on converting one form of energy into another for example mechanical energy into electrical energy and a sensor focuses on detecting and measuring strain or deflection by generating electrical charge or voltage. Also, it's important to realize that not all transducers are sensors but all sensors are transducers since for measuring or detecting some input they do change one form of energy into another.

After discovering piezoelectricity phenomena by Pierre and Jacques Curie in 1880 that certain materials like quartz and tourmaline have the ability to transform mechanical input into elec-

trical output, this opened new opportunities with major breakthroughs in the early 1900s with early piezoelectric transducers composed of quartz and steel sandwich in 1917 used for ultrasonic generation [28] and later on also used to stabilize frequency for broadcasting systems. In recent years, this ability is quite often used for structures for SHM purposes [29, 16]. Piezoelectric sensor that is used in this study is soft PZT which is top left in the figure 2.6. They are used due to their lightweight, inexpensive to use and maintain and also its fast electromechanical response, and applicability for a wide range of frequencies while being also very sensitive to any vibration and reliable. There are other types of piezoelectric sensor such as Polyvinylideen-fluoride (PVDF) sensor with more flexibility but they have been found to be less effective in this research.

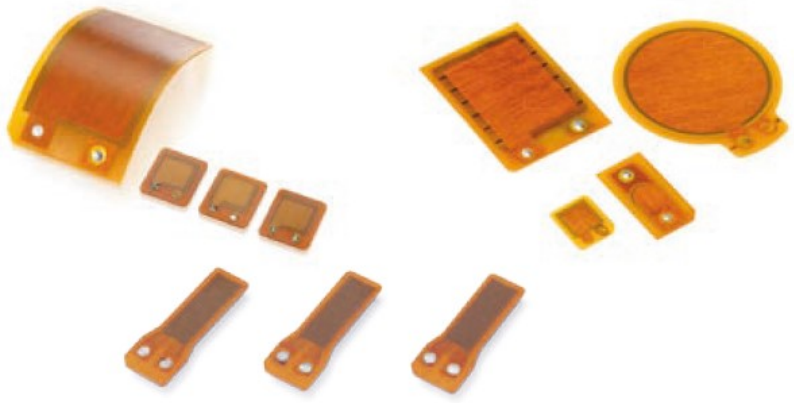
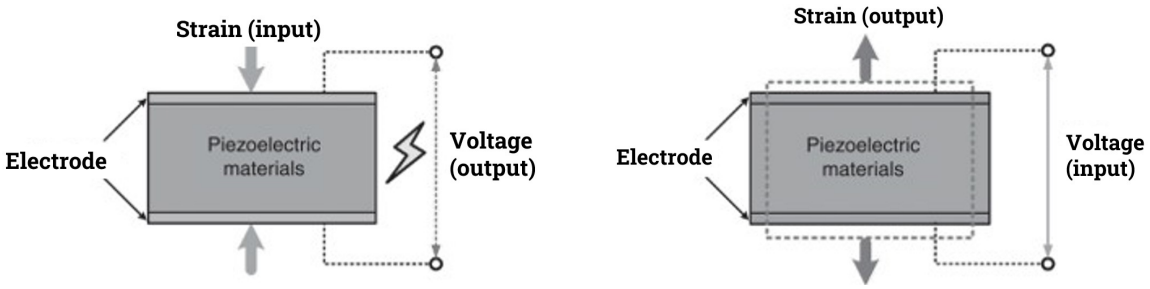


Figure 2.6: Different shapes and sizes of piezoelectric sensor from suplier Physik Instrumente (PI)[30]

2.2.0.1. PZT

Piezoelectric transducers are commonly used for sensing and actuation applications for smart structural systems and structural health monitoring purposes[16]. Some materials like certain types of ceramics, crystals, and other biological matters have the ability of piezoelectricity or piezoelectric effect. PZT or Lead Zirconium Titanate, $Pb(Zr_xTi_{1-x})O_3$ is the most widely used piezoelectric which have perovskite structure with Zr and Ti ions [31] as shown in figure 2.8. This ability enables these materials to generate an alternating current (AC) when subjected to mechanical stress or deformation. Piezoelectric material hence also takes advantage of this method and generates electric charge when squeezed or stretched. This is also known as direct effect of piezoelectricity. Another type is indirect or converse effect of piezoelectricity which is to generate vibration or mechanical force when electrical field or potential is applied. Piezoelectric circuit diagram is illustrated in figure 2.9a and piezoelectric circuit with voltmeter is illustrated in figure 2.9b.



(a) Direct piezoelectric effect

(b) Converse or indirect piezoelectric effect

Figure 2.7: Direct and indirect piezoelectric effect [16]

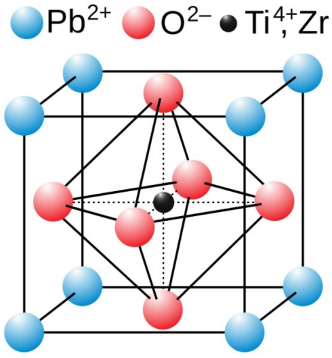
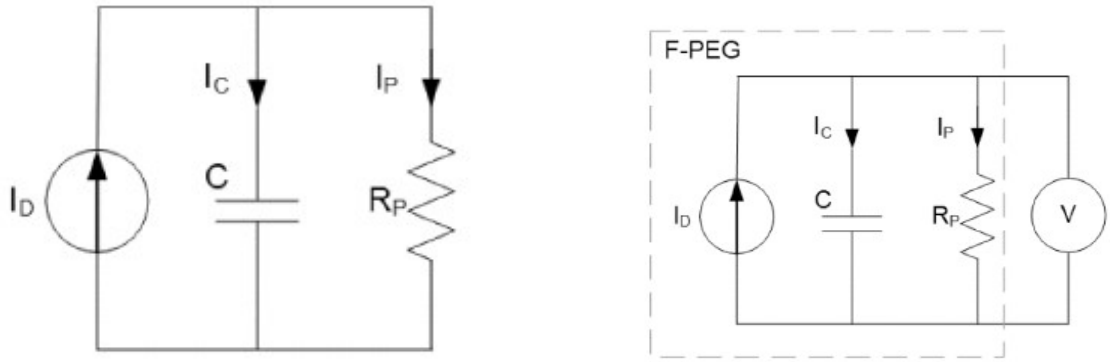


Figure 2.8: PZT crystal [32]



(a) Piezoelectric circuit diagram

(b) Piezoelectric circuit diagram including voltmeter

Figure 2.9: Piezoelectric circuit diagrams

Piezoelectricity can be categorized as shown in figure 2.10 [33] where piezoelectricity is a subgroup of dielectricity where polarization can be changed by mechanical perturbation. It has been well documented that the polarization of these subgroups is not similar as in ferroelectricity spontaneous polarization can also occur eventhough it is also a subgroup of dielectricity. Polarisation is known to be dipole moment per unit volume [34] as shown in equation (2.1). It can be described as charge per unit area as shown in equation (2.2) and it is directly proportional to the applied stress [34] as shown in equation (2.3).

$$P = \frac{\sum \mu}{V} \tag{2.1}$$

where μ is dipole moment and V is volume.

$$P = \frac{Q}{A} \tag{2.2}$$

Where Q is the charge and A is the area.

$$P = d \cdot \sigma \tag{2.3}$$

Where P is polarization, d is piezoelectric coefficient and σ is stress.

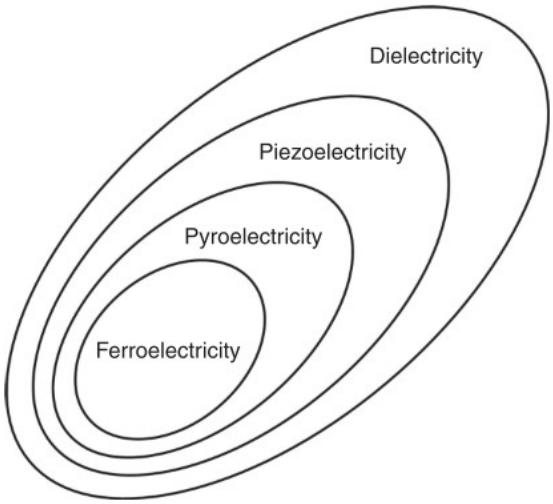


Figure 2.10: Piezoelectricity categorization [33]

Constitutive equations of piezoelectric materials are given in equations (2.4) and (2.5) from IEEE Standard 1987 [35].

$$D_i = \epsilon_{ij}^T E_j + d_{im}^d T_m + \alpha_i \Delta T \tag{2.4}$$

$$S_k = d_{jk}^c E_j + s_{km}^E T_m + \alpha_k \Delta T \tag{2.5}$$

Where D_i is the electric displacement $[C/m^2]$, S_k is the strain (No units), E_j is the applied electric field $[V/m]$ and T_m is the stress $[N/m^2]$, ϵ_{ij}^T is dielectric permittivity $[N/m]$ and s_{km}^E is elastic compliance $[m^2/N]$. Superscript T and E means that they are measured at zero stress and zero electric field. d_{jk}^c and d_{im}^d are the piezoelectric coefficients with units $[m/V]$ or $[C/N]$ The superscripts c and d mean converse and direct piezoelectric effects respectively. ΔT is related to temperature $[^\circ Celsius]$ which is further not considered as this is not investigated in this research. Constitutive equations in matrix form [36] is in equation (2.6).

$$\begin{bmatrix} D \\ S \end{bmatrix} = \begin{bmatrix} \epsilon^T & d^d \\ d^c & s^E \end{bmatrix} \begin{bmatrix} E \\ T \end{bmatrix} \tag{2.6}$$

Where E is the applied electric field matrix and D is the electric displacement. In this research since no electric field is applied, E can be assumed zero then it simplifies to electric displacement = piezoelectric coefficients times stress.

$$D = d_{im}^d T_m \tag{2.7}$$

since piezoelectric is three dimensional and poling directions is mostly at the thickness direction as shown in figure 2.11.

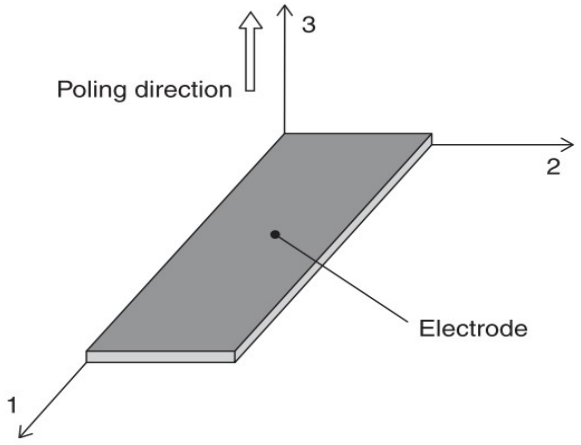


Figure 2.11: Piezoelectric and axes [36]

Applied electric field matrix becomes:

$$E = \begin{bmatrix} E_1 \\ E_2 \\ E_3 \end{bmatrix} \tag{2.8}$$

Electric displacement matrix becomes:

$$D = \begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} \tag{2.9}$$

Other relevant matrices [37] are:

piezoelectric coefficient-matrix:

$$d = \begin{bmatrix} 0 & 0 & d_{31} \\ 0 & 0 & d_{32} \\ 0 & 0 & d_{33} \\ 0 & d_{24} & 0 \\ d_{15} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.10}$$

Compliance matrix:

$$s^E = \begin{bmatrix} S_{11} & S_{12} & S_{13} & 0 & 0 & 0 \\ S_{12} & S_{22} & S_{23} & 0 & 0 & 0 \\ S_{13} & S_{23} & S_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & S_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & S_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & S_{66} \end{bmatrix} \tag{2.11}$$

Permittivity matrix:

$$s^E = \begin{bmatrix} e^{\sigma}_{11} & 0 & 0 \\ 0 & e^{\sigma}_{22} & 0 \\ 0 & 0 & e^{\sigma}_{33} \end{bmatrix} \tag{2.12}$$

Since there is no external electric field applied, equation (2.6) becomes:

$$\begin{bmatrix} D_1 \\ D_2 \\ D_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & d_{15} & 0 \\ 0 & 0 & 0 & d_{24} & 0 & 0 \\ d_{31} & d_{32} & d_{33} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{bmatrix} \tag{2.13}$$

Generated charge is related to electric displacement as shown in equation (2.14). In equation (2.13) σ's are stresses in different directions.

$$q = \int\int \left(\begin{bmatrix} D_1 & D_2 & D_3 \end{bmatrix} \begin{bmatrix} dA_1 \\ dA_2 \\ dA_3 \end{bmatrix} \right) \tag{2.14}$$

Here dA_1, dA_2 and dA_3 are the components of the electrode area in 2-3, 1-3 and 1-2 planes respectively.

So, in general:

$$q = D \cdot A \tag{2.15}$$

Where D is electrical displacement and A is the surface area of the overlapping electrode.

There are different forms of piezoelectric equations like d-form, e-form, g-form and h-form. So equation (2.4) and (2.5) are d-form which relate stress to electric displacement, equation (2.16) and (2.17) are e-form which relate strain to electric displacement. Equations (2.18) and (2.19) are g-form and equations (2.20) and (2.21) are h-forms.

$$D = [\epsilon^S]E + [e]S \tag{2.16}$$
$$T = -[e]^t E + [c^E]S \tag{2.17}$$

$$E = [\beta^T]D - [g]T \tag{2.18}$$
$$S = [g]^t D + [s^D]T \tag{2.19}$$

$$E = [\beta^S]D - [h]S \tag{2.20}$$
$$T = -[h]^t D + [c^D]S \tag{2.21}$$

Where g-form and h-form are extremely rare and also not relevant. E-form is related to this study as strain is one of the determining factors when it comes to asphalt fatigue.

Another approach which is also related to the fundamental equations (2.4) and (2.5) is proposed by Jayant Sirochi and Inderjit Chopra [37]. The voltage output can also be converted to strain according to them as given in equation (2.22) for strain in 1-direction (directions as shown in figure 2.11) [37]:

$$\epsilon_1 = \frac{V_c * C_p}{S_q} \tag{2.22}$$

Where:

$$C_p = \frac{e_{33}^{\sigma} l_c b_c}{t_c} \tag{2.23}$$

Where l_c, b_c, and t_c are the length, width, and thickness of the sensor respectively. e_{33}^{σ} is a member from permittivity matrix as described in matrix (2.12). S_q is sensitivity factor as described in equation (2.24):

$$S_q = d_{31} Y_c l_c b_c \tag{2.24}$$

Where Y_c is the Young's modulus of piezoelectric material.

If we consider Poisson's ratio effect as the sensor is exposed to both longitudinal and transverse effects and also consider the shear lag effect then the final equation becomes:

$$\epsilon_1 = \frac{V_0}{K_p K_b S_q^*} \tag{2.25}$$

Where correlation factor is:

$$K_b = l_{eff} b_{eff} \tag{2.26}$$

l_{eff} and b_{eff} are effective length and width respectively.

Poisson's ratio effect:

$$K_p = (1 - \nu) \tag{2.27}$$

Where ν is Poisson's ratio of the host structure material. Finally, circuit sensitivity is represented by the equation (2.28):

$$S_q^* = \frac{d_{31} Y_c l_c b_c}{C_F} \tag{2.28}$$

Where C_F is feedback capacitance.

These formulas explain the nature of the piezoelectric sensor. They are not used in this study but they were well studied to know how piezoelectric sensor works. An attempt has been made during this thesis to find theoretical formulas to explain flexural tensile strain based on sensor data (voltage amplitude and its frequency) however that was not successful due to the limitation of time and lack of electrical engineering background. Therefore, machine learning is used as an alternative method that not only fits the background of the author but also enables easy estimation of complex problems. This methodology along with the use of machine learning enables any type of sensor to be used with only the difference that the training and testing data of ML models will change.

Another aspect is how piezoelectric is bonded on substrate or host structure. In this research, it is assumed that this shear lag does not interfere with the results since the experiments are done for a short period of time and the sensor is used in the middle of the beam where shear forces are zero for the 4PB test setup which keeps shear lag between the adhesive layer and substrate minimum. Also, the adhesive that is used makes a very strong bond between the sensor and the substrate supported by double-sided tape. The adhesive thickness is very small as the thicker adhesive layer increases the shear lag between the sensor and the adhesive layer.

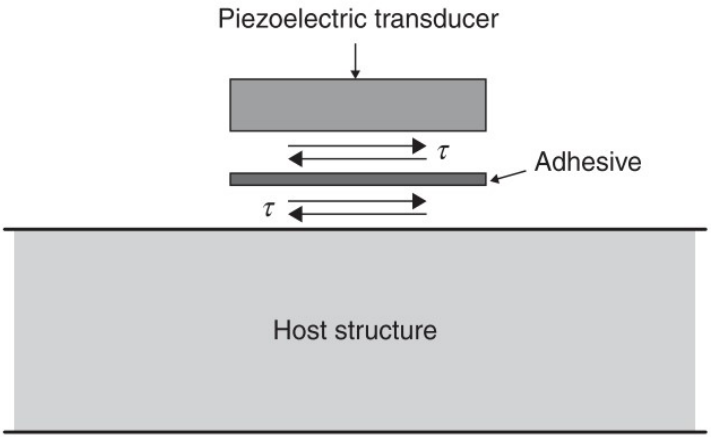


Figure 2.12: Bond between sensor and substrate [16]

2.3. Asphalt

2.3.1. Asphalt failure types

There are many failure phenomena in asphalt as shown in the figure 2.13 and figure 2.14.

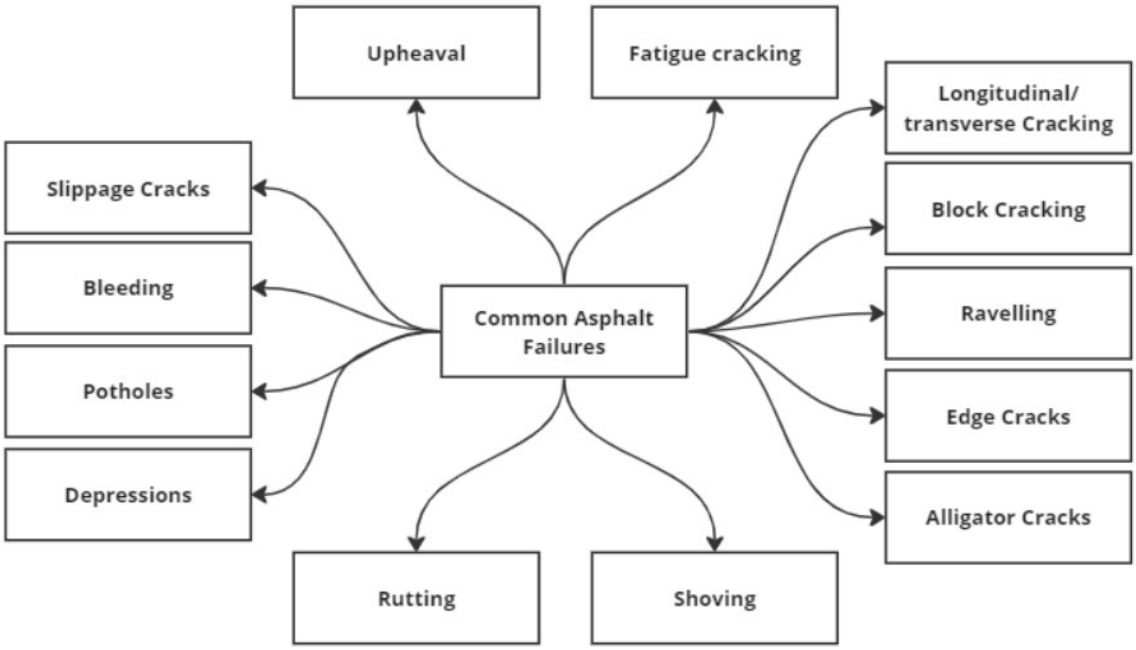


Figure 2.13: Various failure phenomenon in asphalt pavement

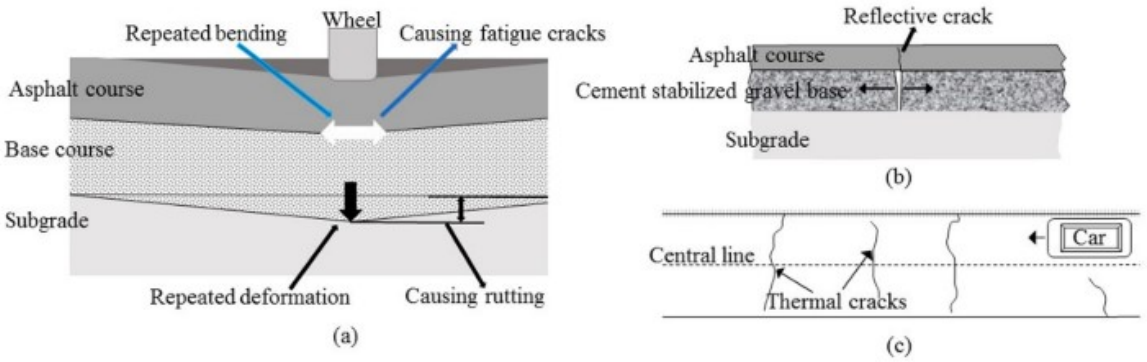


Figure 2.14: Asphalt pavements exhibit a multitude of crack types, encompassing: [38](a) Fatigue cracks; (b) reflective cracks, (c) low temperature-induced shrinkage cracks

Asphalt pavements are susceptible to a variety of crack types, including fatigue cracks, reflective cracks, low-temperature-induced shrinkage cracks, transverse cracks, and others, as depicted in figure 2.13 and figure 2.14. According to Hua-Ping Wang et al. [38] Fatigue cracking predominantly occurs due to improper distribution of heavy traffic loads on the asphalt course, as illustrated in figure 2.14. Inadequate load distribution can lead to fatigue cracks, resembling intricate spiderweb patterns. Additionally, water infiltration or sub-optimal subsurface conditions can also contribute to the development of fatigue cracks [38]. Over time just like any material, the inherent strength of a material also gradually diminishes causing fatigue cracks.

The primary detrimental factors that undermine the functionality of asphalt pavements encompass a range of distresses, such as **cracking** (for example fatigue cracks, top-down cracks, reflective cracks, block cracks, and slippage cracks) or **other** types of distresses such as rutting, potholes, upheaval, corrugation and shoving [38, 39]

Reflective cracks emerge from preexisting cracks in the underlying layers [40], while block cracks exhibit a random distribution of linear fractures. Transverse cracking often arises from seasonal temperature fluctuations, causing shrinkage or the propagation of reflective cracks. Slippage cracks typically result from weak bonding or insufficient adhesion between pavement layers. They are characterized by distinct half-crescent shapes, where the top surface visibly separates from the underlying layers [41]. In figure 2.15, a detailed portrayal of these crack types and distresses can be observed.

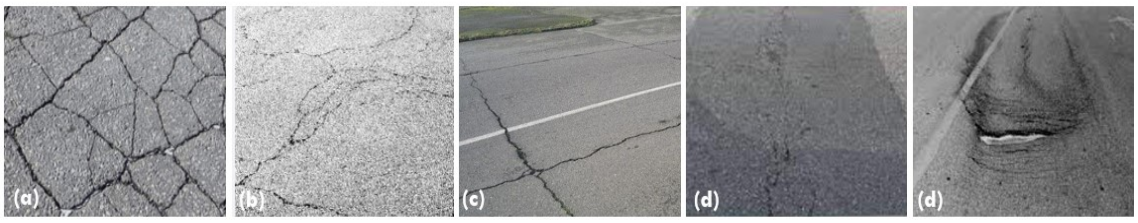


Figure 2.15: Different crack patterns. (a) Fatigue cracks; (b) block cracks; (c) transverse cracks; (d) reflective cracks; (e) slippage cracks

Other type of distresses is shown in figure 2.16.

Figure 2.16: Other types of distresses in asphalt. (a) Rutting; (b) upheaval; (c) potholes; (d) corrugation and shoving

While acknowledging the significance of other failure types and distresses that directly impact the lifespan of asphalt and pavement, this paper specifically concentrates on flexural fatigue resulting from repetitive cyclic loads. Fatigue cracking represents a significant and prevalent form of distress observed in asphalt pavements during their operational lifespan [42, 43, 44, 45, 46]. Fatigue cracking, sometimes referred to as alligator cracking, is a distinct form of pavement deterioration characterized by interconnected cracks that resemble the scales of an alligator as shown in figure 2.15.

This paper reviews and places specific emphasis on the study of flexural fatigue, which arises from the repetitive application of cyclic loads, while also acknowledging the substantial influence of other failure types and distresses that affect the durability of asphalt and pavement structures.

2.4. Fatigue models

Different methodologies and approaches are developed in order to tackle this situation in terms of predicting the fatigue life of new asphalt, and monitoring and predicting damage of asphalt in service. While also developing methodologies to understand the fatigue behavior of asphalt concrete using different laboratory test setups such as four-point bending tests 4PB, 3PB, etc. New methodologies also include the healing of material, its cracking, and its behavior at different temperatures, sizes, etc.

2.4.1. Asphalt fatigue failure

Fatigue cracking, also known as alligator cracking due to its resemblance to the scales of an alligator, is a prominent form of distress observed in asphalt pavements during their operational lifespan [42, 43, 44, 45, 46]. As depicted in figure 2.15, fatigue cracking is characterized by interconnected cracks that progressively deteriorate the pavement. Hence, it is imperative to ensure that the pavement exhibits resilience against fatigue cracking throughout its complete life cycle or service duration.

The asphalt layer, serving as the uppermost component of the pavement structure, bears the brunt of fatigue-induced damage caused by the continuous traffic loading [47, 48, 49]. This damage causes fatigue cracks either from the top or bottom. This damage is influenced by various factors, such as design or installation flaws, excessive cyclic loading, and inadequate drainage systems [50]. Pavement structures are conventionally designed using empirical approaches that rely on comprehensive laboratory and field investigations [51]. Unlike surface cracks, fatigue cracking extends beyond the asphalt layer and affects the entire pavement structure, including its underlying components [52]. The repetitive application of cyclic loads weakens the asphalt, leading to the formation of cracks that propagate both vertically and horizontally.

Mitigating fatigue cracking requires comprehensive and targeted strengthening techniques that address the multifaceted nature of the pavement structure. Advanced engineering approaches play a crucial role in combating this distress. This involves improving design and installation

methods to ensure optimal load distribution and enhance the overall structural integrity of the pavement. Additionally, implementing efficient drainage systems can prevent the accumulation of water, a significant contributing factor to fatigue cracking.

Incorporating durable materials and innovative technologies in pavement construction is another key aspect of fatigue cracking mitigation. Utilizing high-performance asphalt mixtures or integrating innovative additives can significantly enhance the pavement's fatigue resistance and reduce the occurrence of cracks. By adopting proactive measures that encompass improved design, drainage, and material choices, engineers can effectively combat or reduce fatigue cracking and ensure the long-term durability and performance of road surfaces.

In addition to the aforementioned preventive measures and effective design techniques, the inclusion of a robust **fatigue life monitoring system** is imperative in order to manage infrastructure properly. The significance of this system will be further expounded upon in subsequent sections of this paper.

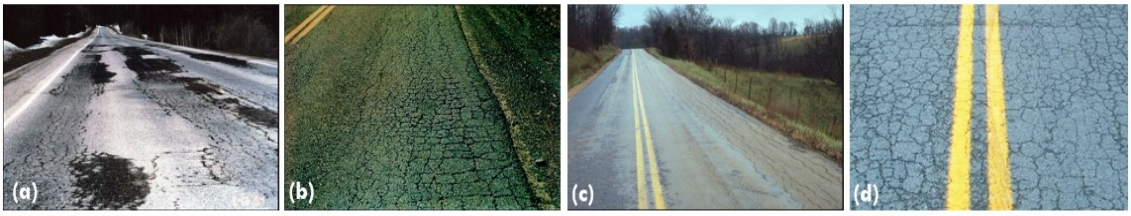


Figure 2.17: Various fatigue cracks [53] due to **(a)** frost; **(b)** load cycles; **(c)** lack of edge support; **(d)** cyclic loads

Fatigue cracking predominantly initiates at the upper layers of thick pavement, as these layers experience the highest stress resulting from the interaction between tires and the pavement surface. This phenomenon, commonly referred to as top-down cracking, is caused by the development of surface tension due to tire forces and the shear stress induced at the edges of the tires [54, 55]. In contrast, in thin pavement structures, fatigue cracking typically initiates at the bottom of the Hot Mix Asphalt (HMA) layer and propagates upwards in the form of longitudinal cracks, a phenomenon known as classical or bottom-up cracking [50, 53]. It is worth noting that fatigue cracks primarily propagate through the bitumen film rather than the aggregate, which is contrary to what would be expected for a composite material [56]. However, when the temperature drops below -10 degrees Celsius, cracks tend to propagate through the aggregate. For fatigue analysis and the prediction of pavement fatigue life, it is recommended to consider temperatures within the range of 15-30 degrees Celsius [57, 54].

When the asphalt pavement is subjected to repeated traffic loads, longitudinal cracks are formed along the wheel paths. After repeated loading, these longitudinal cracks connect to form many-shaped sharp-angled pieces, with a pattern similar to alligator skin. The main causes of alligator cracks on asphalt pavement are excessive traffic loads and improper drainage of the pavement structure.

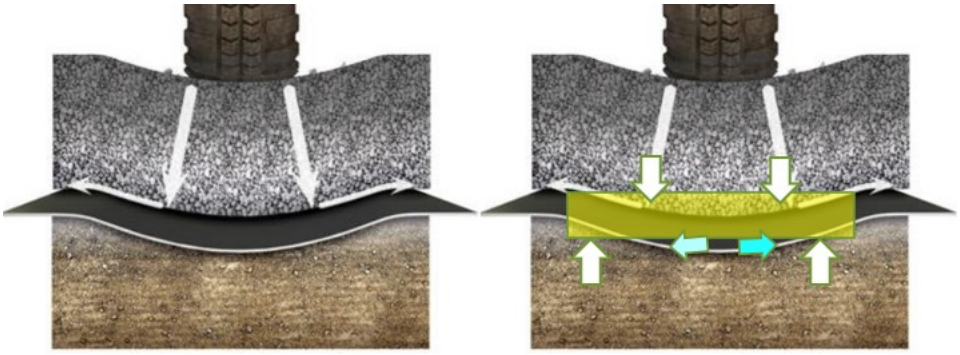


Figure 2.18: Pavement Load Transfer Mechanism under Tire (a) Tire-Pavement Load Transfer (b) Schematic Representation of Loading Similar to 4-Point Bending Model [58]

The performance of the asphalt mixture layer has garnered significant attention from researchers and engineers due to its crucial role in pavement design and durability. Furthermore, the fatigue resistance of asphalt layers plays a critical role in determining the appropriate pavement thickness. In order to obtain a dependable pavement design output, it is essential to accurately evaluate the fatigue performance of asphalt mixtures. As a result of these considerations, the assessment of fatigue performance on asphalt layers has become a focal point in the field of pavement research.

In the following sections, we will explore various test setups and methodologies, highlighting the importance of load transfer in fatigue testing and its impact on assessing the structural performance of asphalt pavements. It is worth noting that traditional empirical design methods have become inadequate due to changes in several factors, such as increased traffic loads, higher traffic volumes, the use of recycled materials, and the effects of climate change [59, 54]. Consequently, engineers have developed new approaches to ensure reliable pavement design, such as Performance-Related Specifications (PRS) based on mechanistic principles. These approaches utilize the Performance Volumetric Relationship (PVR) to predict the service life of asphalt mixtures based on their volumetric properties [54]. Various parameters and factors influence the fatigue life of asphalt concrete, leading to the use of different failure criteria, including indicators based on stiffness, energy, stress degradation ratio, and more.

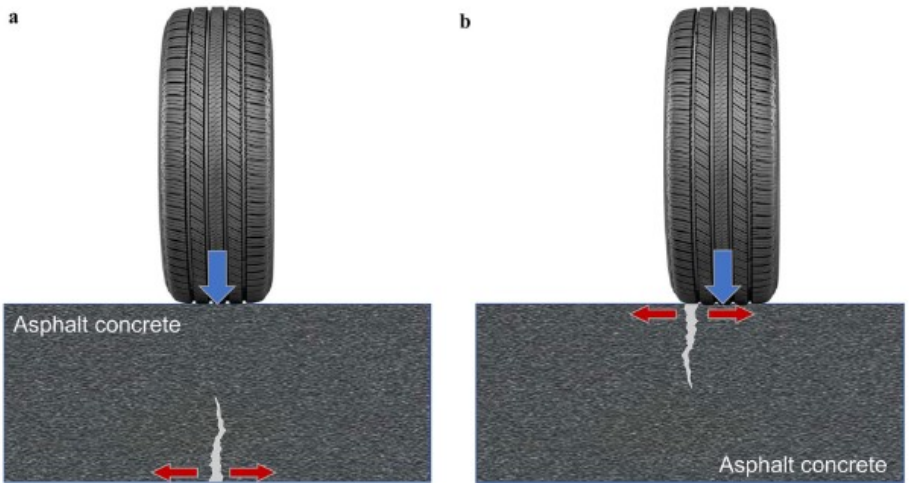


Figure 2.19: Fatigue cracking: a bottom-up cracking and b top-down cracking [60].

2.4.2. Failure models

All the models in this chapter have been classified and presented in the figure 2.21.

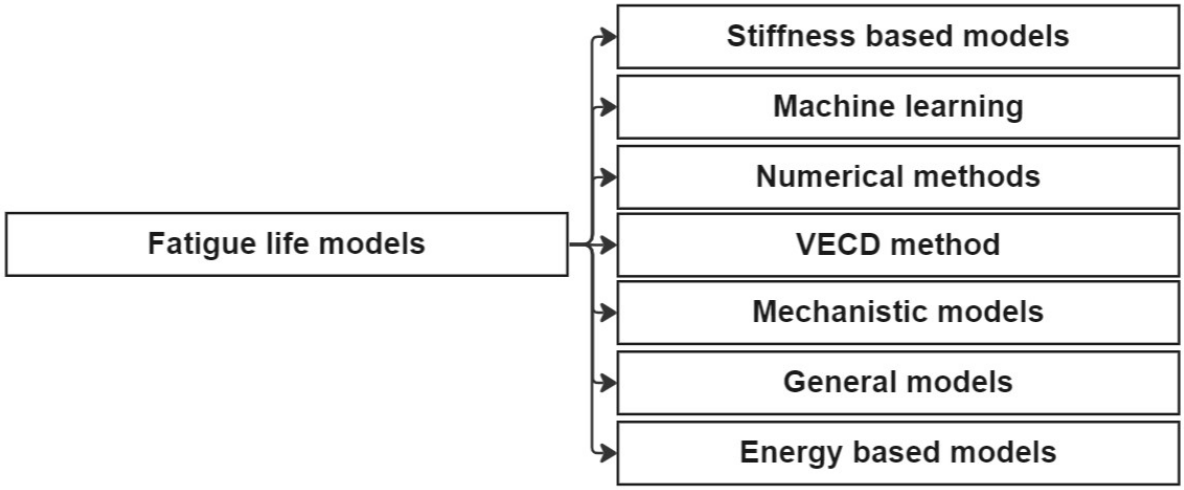


Figure 2.20: Different types of asphalt fatigue models

These models, representing different categories, aim to aid readers by offering asphalt fatigue models and delivering thorough and vital descriptions for each. Some of these models have been previously discussed in other review papers, but they are also reiterated here to provide a clearer and more comprehensive explanation.

2.4.2.1. Fatigue life models

This chapter describes and reviews various fatigue analysis methods used to estimate and predict fatigue life. This chapter provides a comprehensive overview of fatigue life prediction, with a particular focus on phenomenological models. Subsequent (sub)chapters delve into various alternative approaches, including the energy method, mechanistic models, numerical simulations, and VECD (Viscoelastic Continuum Damage) methods. By exploring these diverse methodologies, a holistic understanding of fatigue life prediction is achieved, enabling engineers to employ a range of tools and approaches to accurately assess and forecast the fatigue performance of asphalt pavements. These methodologies encompass a range of approaches that are rooted in the principles of stiffness modulus, energy considerations, viscoelastic continuum mechanics, and other models that account for cumulative damage or fatigue life. Fatigue models can be broadly classified into two types: phenomenological and mechanistic approaches. Phenomenological models are based on empirical relationships and observations, while mechanistic approaches involve a deeper understanding of the underlying physics and mechanics of fatigue behavior. These models aim to predict the fatigue life of asphalt based on factors such as loading conditions, material properties, and environmental effects. By employing different modeling techniques, engineers and researchers can gain insights into the fatigue mechanisms and develop more accurate predictions for the performance of asphalt pavements.

Fatigue life models elucidate the correlation between the fatigue life of asphalt concrete and the levels of strain/stress (also known as the phenomenological approach), both with and without considering additional influential factors such as temperature, stiffness modulus, rest periods, moisture content, and repetitive cyclic loading. These models play a vital role in road design and have consequently been incorporated into pavement structural design codes and specifications (NEN codes, [61, 62, 8]). Furthermore, these models are instrumental in the maintenance of roads, which includes the implementation of structural health monitoring [17, 15, 19, 20].

It is noteworthy that in the field of predicting asphalt fatigue life, various modeling approaches are employed. Phenomenological models, which encompass Miner's cumulative damage principle and similar techniques, are commonly utilized. These models take into account the progressive accumulation of damage due to repeated loading cycles. An alternative approach to

fatigue modeling is rooted in fracture mechanics, involving the calculation of cumulative crack length for estimating fatigue life. Additionally, continuum damage models assess the extent of damage within asphalt concrete layers and use transfer functions to quantify this damage as a percentage of the crack area. The integration of these diverse modeling methodologies enhances our understanding of fatigue behavior and enables more precise predictions of asphalt pavement fatigue life [54].

All the models in this chapter have been summarized and ranked according to publish date in the figure 2.21.

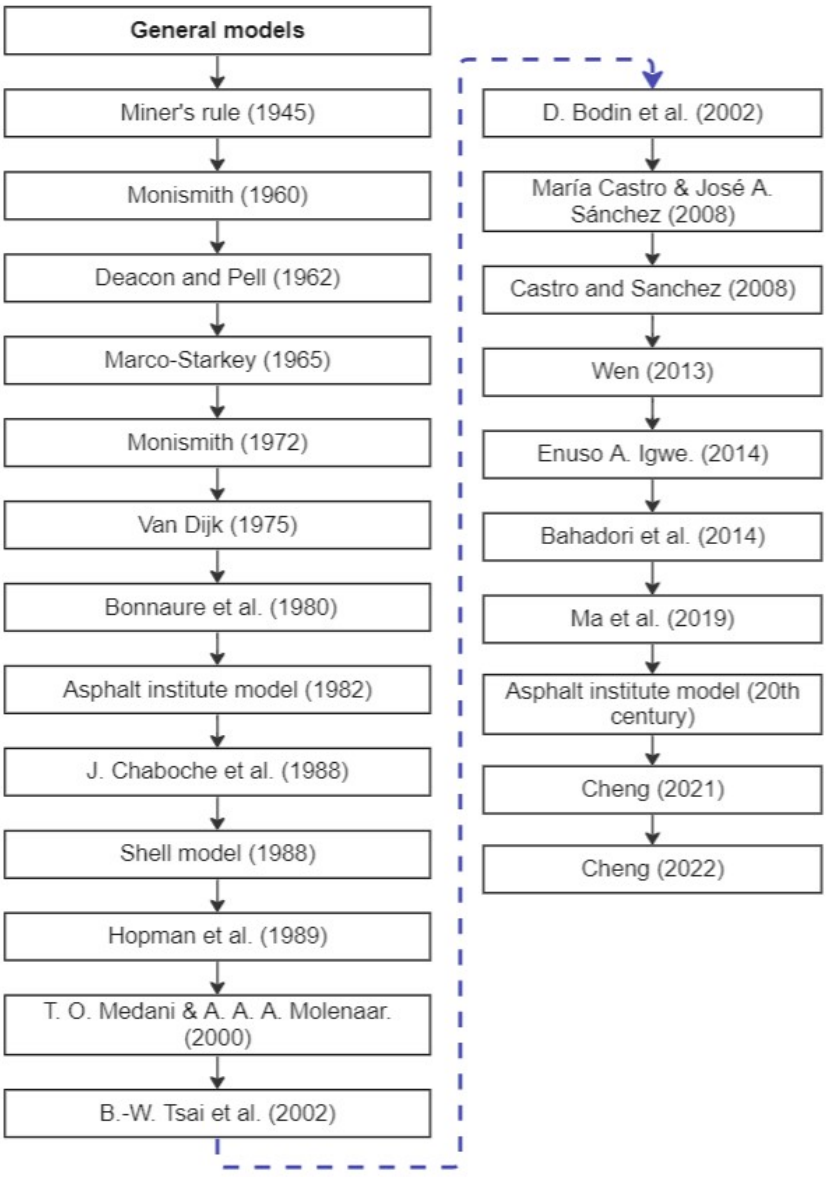


Figure 2.21: Summary of general models ranked by publish date

Classical fatigue models for asphalt concrete were originally developed in the early 1960s by Monismith [63] and Deacon and Pell (1962) [64]. These models are as follows[8]:

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \tag{2.29}$$

$$N_f = C \cdot \left(\frac{1}{\sigma_t}\right)^D \tag{2.30}$$

In these models, the fatigue life of the asphalt mixture (N_f) is determined by the strain level (ϵ_t), stress level (σ_t), and fitting parameters A, B, C, and D.

The provided models describe the fatigue life behavior and trends in both controlled strain and controlled stress tests, respectively. However, Monismith later proposed an updated model that incorporates the stiffness modulus parameter, aiming to enhance the prediction accuracy of fatigue life for asphalt mixtures.

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \cdot \left(\frac{1}{S}\right)^C \tag{2.31}$$

In these models, the fatigue life of the asphalt mixture N_f is determined by the strain level ϵ_t, the initial stiffness modulus of the mixture S, and the fitting parameters A, B, and C.

In 1975, Van Dijk developed a fatigue life model that incorporates the phase angle of the asphalt mixture. This model is represented by Equation (2.32).

$$N_f = A \cdot \left(\frac{1}{\epsilon_t \cdot S \cdot \sin\delta}\right)^B \tag{2.32}$$

In the developed fatigue life model for asphalt mixture by Van Dijk, the fatigue life N_f is determined by the strain level (ϵ_t), the initial stiffness modulus of the mixture S, the phase angle of the mixture δ, and the fitting parameters A and B.

As demonstrated earlier, temperature has a significant impact on determining the fatigue life of asphalt mixtures. It has been substantiated that Equation (2.33) is a practical approach to quantify the influence of temperature on fatigue life, irrespective of the type of mixture and test conducted [65].

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \cdot e^{CT} \tag{2.33}$$

In equation (2.33), the fatigue life of the asphalt mixture N_f is influenced by the strain level ϵ_t, the temperature (T), and the fitting parameters A, B, and C. This equation provides a practical means to characterize the impact of temperature on fatigue life, regardless of the specific mixture type or test conducted.

In 2022, Cheng made further improvements to Equation (2.33) by enhancing its ability to account for rest periods and incorporating both temperature and strain levels. These advancements allow for a more comprehensive and accurate characterization of the influence of various factors on fatigue life in asphalt mixtures. This equation is illustrated as follows:

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \cdot e^{(CT+D\cdot RP)} \tag{2.34}$$

In Cheng's improved version of equation (2.34) [5], the fatigue life of the asphalt mixture (N_f) is determined by the strain level (ϵ_t), temperature (T), and rest period (RP), along with the fitting parameters A, B, C, and D. This enhanced equation takes into account the combined effects of temperature, strain level, and rest periods to provide a more comprehensive understanding of fatigue life in asphalt mixtures.

In addition, there exist, various other phenomenological models, including the well-known model proposed by the Asphalt Institute [66, 67, 68, 69].

$$N_f = k_1 \epsilon_t^{k_2} E_0^{k_3} \tag{2.35}$$

Another model is [70]:

$$N_f = 10^{2.3^{-1} Ln(a(\epsilon_t)^b (E^*)^c} \tag{2.36}$$

Where E^* is the dynamic modulus. a,b and c are the fitting parameters.

Additionally, there exist fatigue life models that incorporate fracture mechanics principles and the properties of the asphalt mixture. For instance, Wen proposed a model in 2013 that utilizes Fracture Work Density (FWD), as depicted in equation (2.37). Similarly, Bahadori et al. introduced a model in 2014 that incorporates Fracture Energy (FE), as illustrated in equation (2.38). These models provide a framework to assess fatigue life by considering the fracture mechanics characteristics of the asphalt mixture.

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \cdot FWD^C \cdot h^D \tag{2.37}$$

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \cdot FE^C \tag{2.38}$$

In the fatigue life models proposed by Wen (2013) [71] and Bahadori et al. (2014) [72], the fatigue life of the asphalt mixture (Nf) is determined by the strain level ϵ_t, the fracture work density (FWD) in the case of Wen's model, or the fracture energy (FE) in the case of Bahadori et al.'s model. These models incorporate fitting parameters A, B, C, and D to establish the relationship between these factors and the fatigue life behavior of the asphalt mixture.

Furthermore, there exist models that take into account the damage state of asphalt mixtures or the cumulative damage they have experienced. Notable examples include the model developed by Castro and Sanchez in 2008 [73], as shown in Equation (2.39), and the model proposed by Ma et al. in 2019 [74], as illustrated in Equation (2.40). These models are particularly valuable as they consider the damage history and current damage state of the mixture, thereby providing a more comprehensive assessment of fatigue life.

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \cdot D^C \tag{2.39}$$

$$N_f = A \cdot \left(\frac{1}{\epsilon_t}\right)^B \cdot (1-D)^C \tag{2.40}$$

In the fatigue life models developed by Castro and Sanchez (2008) and Ma et al. (2019), the fatigue life of the asphalt mixture N_f is determined by the strain level (ϵ_t), the damage of the mixture D, and the fitting parameters A, B, and C. These models take into account the influence of the damage history and current damage state of the asphalt mixture on its fatigue life.

The Miner's rule [75], developed around 1945, assumes a linear accumulation of fatigue damage with loading cycles, as shown in Equation (2.41). Despite its wide adoption due to its computational simplicity, it has been observed that linearity only applies during the intermediate stage of the test. Fatigue damage follows a nonlinear pattern during the initial and final stages of loading repetition.

$$D = \frac{N}{N_f} \tag{2.41}$$

Figure 2.22 depicts a typical curve representing Miner's criteria.

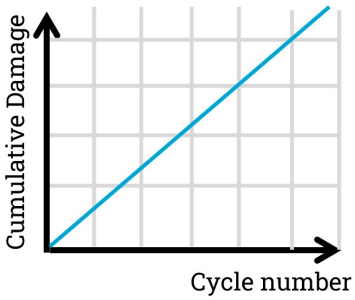


Figure 2.22: Contribution to damage according to Miner's hypothesis

The curve appears as a straight line because the contribution to damage from each load cycle is consistent. However, it is important to note that the actual formation and growth of cracks may have a different lifespan compared to what is predicted by Miner's criteria.

Hopman et al. (1989) [76]introduced a nonlinear model based on Miner's rule that incorporates an exponent, as depicted in equation (2.42). Another nonlinear fatigue damage law, described by Equation (2.43), incorporates linear functions of strain amplitude.

$$D = (\frac{N}{N_f})^x \tag{2.42}$$

In this context, "D" represents the fatigue damage caused by repetitive loading, "N" represents the number of loading cycles, and "N_f" represents the fatigue life of the mixture. The parameter "x" is an exponent that falls within the range of 0.82 to 0.92.

The proposed model from Hopman was also introduced by Marco-Starkey in 1965 and by Manson Halford. It defines the damage function for variable x as follows:

$$a = f(\sigma, t) = 1 + \left(\log \frac{\sigma}{\frac{1}{2}\sigma_s}\right)^t \tag{2.43}$$

Here, t is a material parameter. Substituting this into equation (2.41), we get:

$$D = \left(\frac{n}{N}\right)^{1+\left(\log \frac{\sigma}{\frac{1}{2}\sigma_s}\right)^t} \tag{2.44}$$

In this equation, t is still related to the material parameter, "D" represents the damage, and σ denotes the stress.

Damage parameter can also be calculated by examining the relative loss of modulus between the initial state and the current state at a certain cycle N. The calculation is given by:

$$D = \frac{|E_0^*| - |E^*|}{|E_0^*|} \tag{2.45}$$

Here, $|E_0^*|$ represents the norm of the initial state complex modulus, and $|E^*|$ denotes the complex modulus norm at cycle N.

A proposed damage model is given by [77]:

$$D = \left(\frac{n}{N}\right)^{1+\left(\log \frac{\sigma}{\frac{1}{2}\sigma_s}\right)^t} \tag{2.46}$$

In this equation, N represents the number of cycles, ϵ_0 is the initial strain, and a, b, and c are fitting parameters.

However, this method has a disadvantage. The number of data points obtained from the experiment is proportional to the test duration, resulting in lower ϵ_0 having a greater influence on the estimated regression parameters.

To address this, another model is proposed, as described in [77]. The model is given by:

$$N = \beta D^c \tag{2.47}$$

Here, N is the number of cycles, D represents the damage, and β and c are constants to be determined. The procedure for finding the parameters and getting fatigue curves is illustrated in the following figure:

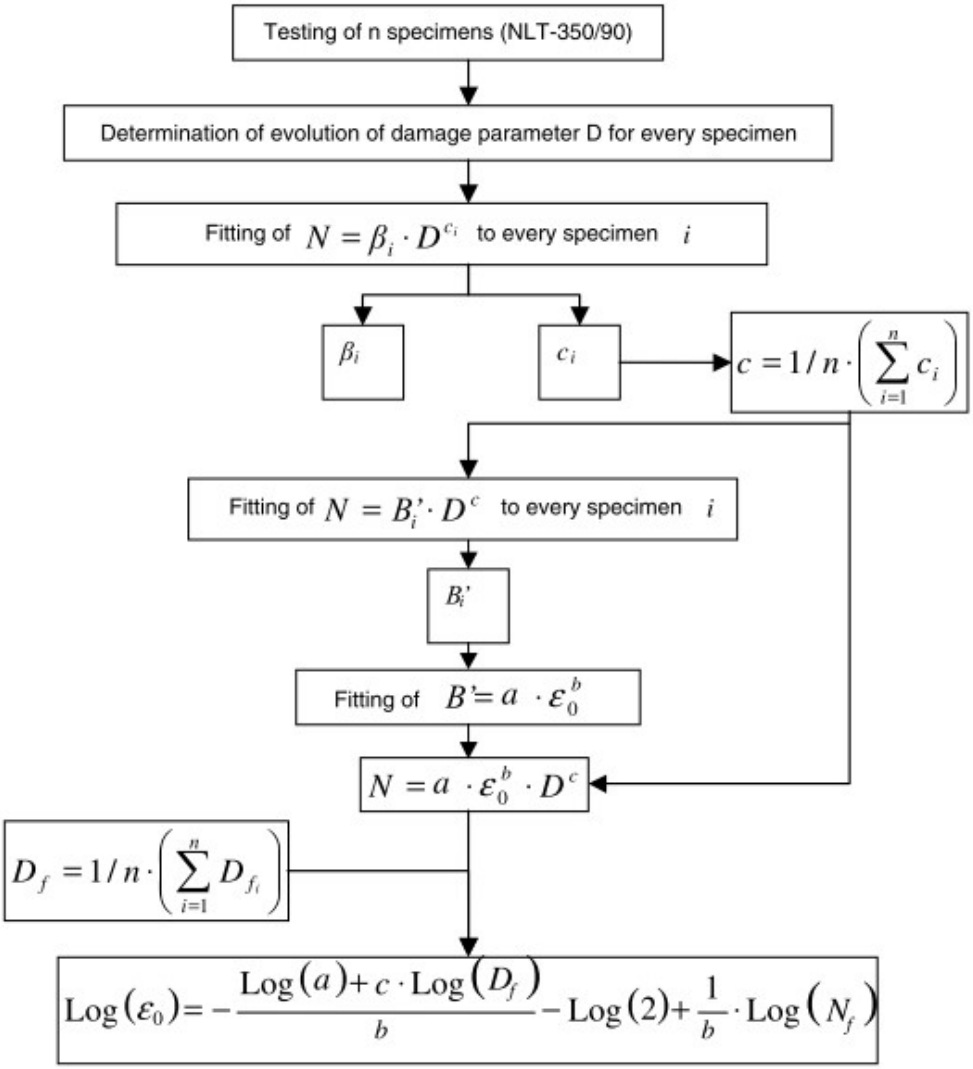


Figure 2.23: Fatigue curve estimation method [77]

According to Yingjun Mei et al. [78], an alternative approach for predicting fatigue life involves cumulative damage. The method comprises several steps outlined in their study:

1. The establishment of a strain fatigue equation.
2. Determination of the cumulative number of standard axle loads, denoted as N_e. This parameter is calculated by summing the individual contributions from each temperature range, represented as N_{iS}:

$$N_e = \sum_{i=1}^{5} N_{iS} \tag{2.48}$$

   Here, i corresponds to the different temperature ranges.
3. Calculation of the maximum tensile strain, denoted as ϵ_{ji}.
4. Evaluation of the fatigue damage degree, denoted as FDD_i, using the following formula:

$$FDD_i = \frac{N_{iS}}{\frac{N_{iy}}{1.3}} \tag{2.49}$$

   Here, N_{iy} represents the cumulative number of standard axle loads within the design life of the pavement.

5. Summation of all the FDD values yields the fatigue damage degree of the asphalt pavement throughout its service life.
6. The fatigue life can be estimated using the formula:

$$Fatigue\ life = \frac{1}{FDD} \cdot \sum_{i=1}^{n} N_{iS} \tag{2.50}$$

Here, n signifies the total number of temperature ranges, and N_{iS} denotes the cumulative number of standard axle loads within each temperature range in the design life of the pavement.

Another fatigue damage law, which is nonlinear in nature, describes the connection between damage and the number of loading cycles. This relationship has been documented in previous research studies and is shown in the following equation (2.51)([79]).

$$\frac{dD}{dN} = B \cdot \frac{(A(1-D))^{(B-1)/B}}{A^{B-1}} \tag{2.51}$$

In this context, "D" represents the fatigue damage resulting from cyclic loading, "N" denotes the loading cycle, and "A" and "B" are linear functions of the strain amplitude.

Additional nonlinear fatigue damage models, such as those represented by Equations (2.52) and (2.53), exist, incorporating fitting parameters to analyze damage accumulation processes and fit nonlinear trends in fatigue damage.

Chaboche and Lesne [80]

$$D = 1 - (1 - \frac{N}{N_f}^{\frac{1}{1-\alpha}})^{\frac{1}{\beta-1}} \tag{2.52}$$

Bodin et al [81]

$$D = (\frac{C(1-\alpha)}{1+\beta} \cdot (\epsilon_\alpha)^{\beta+1} N)^{\frac{1}{1-\alpha}} \tag{2.53}$$

where D is fatigue damage, N is the loading cycle, N_f is the fatigue life of the mixture,α, β, C are fitting parameters.

Overall, these models provide valuable tools for analyzing the fatigue life and damage evolution of asphalt mixtures, considering various factors and their interplay.

In the given fatigue life models for asphalt mixtures, the fatigue life N_f is influenced by the strain level ϵ_t, the damage of the mixture (D), and the fitting parameters A, B, and C. These parameters are used to establish the relationship between the strain level, damage, and the resulting fatigue life of the asphalt mixture.

In addition to external parameters related to loading conditions, there are also models that consider internal parameters or factors associated with asphalt mixture design such as asphalt binder and mixture properties. One such model was developed by Bonnaure et al. in 1980 [82] and is represented by equation (2.54). This model incorporates internal parameters and design-related factors to enhance the accuracy of fatigue life predictions for asphalt mixtures. Notably, this model introduces two material-related parameters, namely the penetration index of the asphalt binder and the asphalt content of the mixture, to enhance its predictive capability for fatigue life estimation.

$$N_f = (A \cdot PI + B \cdot PI \cdot \nu_b + C \cdot \nu_b + D) \cdot \left(\frac{1}{\epsilon_t}\right)^E \cdot \left(\frac{1}{S}\right)^F \tag{2.54}$$

Shell has also developed methods of estimating fatigue resistance of asphalt mixtures [68]. These models have been developed after testing many beams on wide range of temperature and other parameters. These models are as follows:

Stress controlled mode:

$$N_f = (0.0252PI - 0.00126PIV_b + 0.00673V_b - 0.0167)^5 \epsilon_t^{-5} S_m^{-1.4} \tag{2.55}$$

strain controlled mode:

$$N_f = (0.0252PI - 0.0085PIV_b + 0.04543V_b - 0.112)^5 \epsilon_t^{-5} S_m^{-1.8} \tag{2.56}$$

where N_f is a number of cycles to failure, PI is the penetration index, V_b is the volume of the binder, ϵ_t is the initial tensile strain and S_m is mixture stiffness.

Medani and Molenaar proposed a model [83] that utilizes the Wohler equation to estimate the number of cycles leading to failure. Their model incorporates important fracture mechanics parameters, such as the exponent n. The specific models proposed are as follows:

$$N_f = k1(\frac{1}{\epsilon_t})^n \tag{2.57}$$

$$n = \frac{2}{m(0.541 + \frac{0.346}{m} - 0.03524V_a)} \tag{2.58}$$

$$\log(k_1) = 6.586 - 3.762n + \frac{3209}{S_m} + 2.332 \log(V_b) + 0.149\frac{V_b}{V_a} 0.928PI - 0.0721T_{R\&B} \tag{2.59}$$

where k_1 is the coefficient which was obtained after 108 strain-controlled flexural beam fatigue tests. 'm' is the stiffness master curve slope, $T_{R\&B}$ is the softening point for the binder determined by the ring and ball test (°Celsius). S_m is the mixture stiffness in MPa, and 'n' is the fracture parameter [84].

Another model is developed by asphalt institute [85] which is shown in equation (2.60) for asphalt mixture with 11% binder volume and 7% air void content. equation (2.61) and (2.62) can be used to find correction factors for other type of volumetric properties.

$$N_f = A \csc 0.00432C\epsilon_t^{-3.291}|E^*|^{-0.854} \tag{2.60}$$

$$C = 10^M \tag{2.61}$$

$$M = 4.84(\frac{V_{beff}}{V_a + V_{beff}} - 0.69) \tag{2.62}$$

where $|E|^*$ is mix stiffness dynamic modulus, A is 18.4 FSF, C and M are volumetric factors, V_{beff} effective binder volume % and V_a is air void content % [84].

A similar model is developed by the University of California [86] used in the analysis of the paper in the WesTrack experiment [84].

For fine mixes:

$$ln(N_f) = -27.0265 - 0.1439V_a + 0.4148P_{Wasp} - 4.6894ln(\epsilon_i) \tag{2.63}$$

for fine-plus mixes:

$$ln(N_f) = -27.3409 - 0.1431V_a + 0.4219P_{Wasp} - 0.0128ln(T) - 4.6918ln(\epsilon_i) \tag{2.64}$$

For coarse mixes:

$$ln(N_f) = -27.6723 - 0.0941V_a + 0.6540P_{Wasp} - 0.0331T - 4.5402ln(\epsilon_i) \tag{2.65}$$

Where N_f is predicted fatigue life, V_a is air void content (%), P_{Wasp} asphalt content (% by weight), T is the temperature (at 150mm deep in degree celsius) and ϵ_t is maximum principal tensile strain.

The aforementioned models provide insight into the intricate nature of asphalt mixtures and the various factors that influence fatigue life. These factors include strain/stress levels, temperature, stiffness modulus, rest periods, and mixture design parameters such as asphalt binder properties and asphalt content [8]. Analyzing the results of these models and considering the parameters, it is observed that the fatigue life of asphalt mixtures tends to increase with longer rest periods, higher fracture energy, and greater fracture work density. Conversely, fatigue life decreases with elevated stress or strain levels and increased existing damage [8].

2.5. Fatigue test methods

The development of reliable test methods to assess the fatigue performance of asphalt layers is of great importance. Researchers and scholars have devoted significant efforts to devise a range of methods, including full-scale field tests, field surveys, and laboratory fatigue tests. These testing approaches enable a comprehensive evaluation and analysis of asphalt, providing engineers with valuable insights into the behavior and properties of asphalt layers. By conducting these tests, engineers can better understand the material and predict its performance in real-world conditions [87, 88, 89, 90, 91, 92, 93]

It is important to acknowledge that asphalt is a viscoelastic and anisotropic material, implying that its mechanical properties are highly influenced by various factors, including temperature, loading frequency, loading mode, loading location, rest periods, and more. Consequently, the intricate nature of asphalt necessitates a thorough comprehension of its behavior across different conditions As a result, the fatigue responses of the asphalt mixture are influenced by the specific loading scenarios to which it is subjected [87, 94, 95]. Different loading conditions impact the fatigue behavior of asphalt mixtures. Static, dynamic, and cyclic loads, as well as variations in strain or stress amplitudes, have distinct effects on the material's durability. Understanding and analyzing these various loading scenarios are crucial for accurately assessing the fatigue performance and durability of the asphalt mixture in practical applications.

To accurately replicate the fatigue behavior of asphalt mixture under real field conditions, laboratory fatigue tests include carefully devised loading setups [87]. These experimental configurations are meticulously devised to replicate the loading conditions encountered by asphalt mixtures in real-world scenarios. Parameters such as loading frequency, magnitude, and mode are carefully incorporated to establish a testing environment that closely emulates actual fatigue loading scenarios. This meticulous approach guarantees that laboratory fatigue tests yield dependable and indicative outcomes, enabling enhanced comprehension of the fatigue performance of asphalt mixtures in practical applications. The objective of this paper is to provide a comprehensive review and introduction of multiple aspects related to the measurement of fatigue response in asphalt. This includes an examination of various test methods, fatigue life prediction models, fatigue life monitoring systems, and recent advancements in fatigue life monitoring. The objective is to provide a broad coverage of various subjects and offer a comprehensive summary of the current research and advancements in assessing the fatigue performance of asphalt. Additionally, the review will include a synthesis of previous literature reviews, ensuring a comprehensive overview of the topic.

Numerous models have been proposed to characterize the fatigue performance of asphalt mixtures. Some of these models are based on phenomenological fatigue life models [64, 63, 96, 65, 97, 75] , which describe the relationship between fatigue life and various factors. Other models focus on the evolving patterns of the stiffness modulus [98, 99, 100, 101, 102, 103, 98], aiming to analyze the performance attenuation tendency of the mixture. Also, some models quantify the accumulation of damage [80, 76, 81, 79, 75, 77, 80, 81].

Alternative approaches based on energy and viscoelastic continuum damage (VECD) have been developed to overcome the limitations of earlier methods and achieve a unified fatigue law for asphalt mixture that is unaffected by loading conditions[104, 105, 106, 107, 108, 109] . Laboratory test results are extrapolated to field performance through the application of a shift factor. This factor accounts for various factors including aging, healing, traffic density, temperature, climate, moisture damage, service life, and more. Typically, the shift factor used is 15 to 20 times the value obtained in laboratory testing, enabling a more accurate representation of real-world conditions.

2.5.1. Loading modes

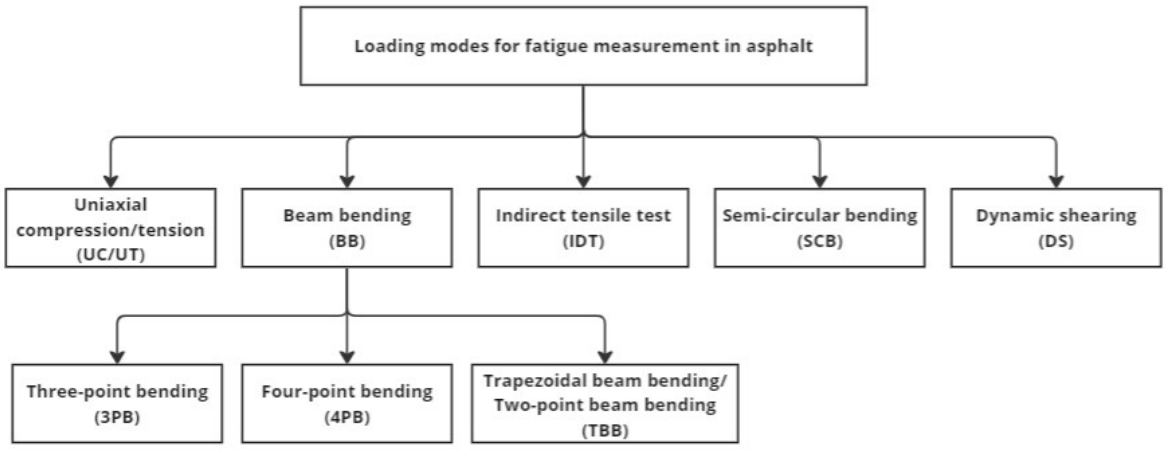


Figure 2.24: Test types & loading modes for evaluating fatigue life

Under loading mode, fatigue tests can be categorized into three groups: (1) Simple flexure, (2) direct uniaxial, and (3) diametral load tests [**review1**]. These tests provide valuable insights into the fatigue behavior of materials and help in evaluating their performance under different loading conditions. In figure 2.24 different loading modes to measure the fatigue response of asphalt have been shown which are uniaxial compression/tension (UC/UT), beam bending (BB), indirect tensile test (IDT), semi-circular bending (SCB), dynamic shearing (DS) [8, 54, 110, 111, 112, 113]. The pictures [114] and schematic diagrams [8] of test setups and loading modes for measuring fatigue response in asphalt are illustrated in figure 2.25.

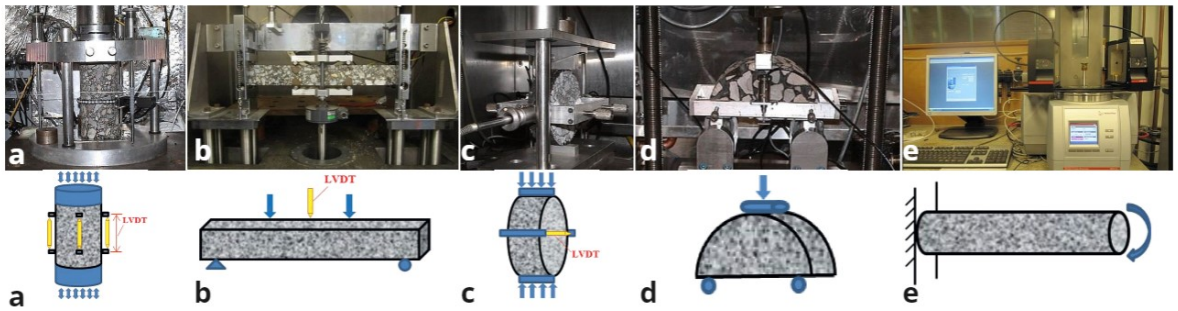


Figure 2.25: Pictures [114] and schematic diagrams [8] of test types & loading modes for evaluating fatigue life (a) uniaxial compression/tension (UC/UT); (b) beam bending (BB); (c) indirect tensile test (IDT); (d) semi-circular bending (SCB); (e) dynamic shearing (DS)

Table 2.2: Loading modes for fatigue measurements and its relevant specifications and references

Loading mode	Specification(s)/reference(s)
UC/UT	NEN-EN 12697-25 AASHTO (2018 [115]) AASHTO (2019 [116])
BB	NEN-EN 12697-24 AASHTO (2017 [117],2014 [101]) ASTM (2010 [118]) CEN (2018 [119]) Afnor(1992) [120] BSI(2018 [121, 122])
IDT	NEN-EN 12697-24 CEN (2018 [119]) AASHTO (2016 [123],2018 [124]) Cheng 2021 [65] BSI(2018) [121, 122]
SCB	NEN-EN 12697-44:2019 AASHTO (2016 [123]) CEN (2018 [119])
DS	NEN-EN 12697 AASHTO (2014 [101]),

Controlled loading can be categorized into two types: controlled stress and controlled strain. In constant stress loading, the stress remains constant while the strain increases, while in constant strain loading, the strain remains constant while the stress decreases. For beam thicknesses less than 50 mm, it is advisable to use constant strain loading. However, for other thicknesses, a combination of constant stress and strain modes can be employed [51]. These different loading modes provide flexibility in testing and allow for a more comprehensive evaluation of the fatigue behavior of asphalt materials.

In the provided figure 2.26, we can observe the representation of a constant stress test for an asphalt mix. The constant stress is maintained throughout the test, as per the user's input and requirements. As the number of cycles increases under this constant stress condition, the strain in the asphalt mix gradually increases. This can be attributed to the decrease in the stiffness of the asphalt concrete over time and with an increasing number of cycles.

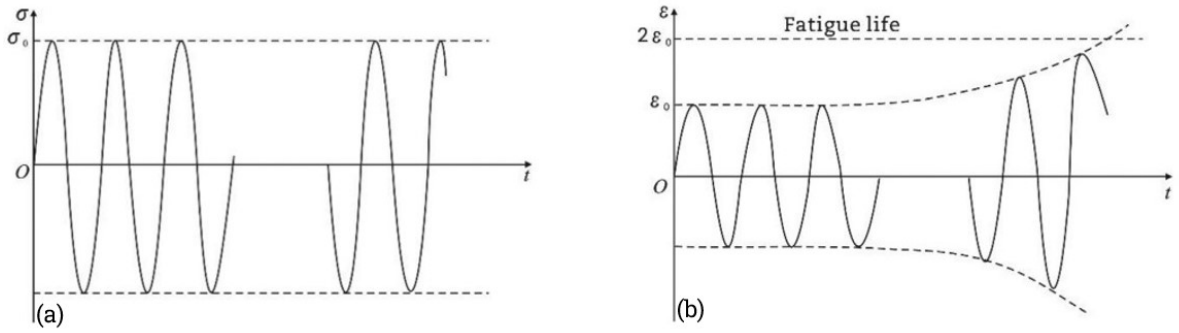


Figure 2.26: Constant stress test [54] (a) stress; (b)strain

In the context of a constant strain test for an asphalt mix, the behavior is slightly different compared to the constant stress test. In the figure 2.27, we can observe the representation of a constant strain test. In this test, the strain applied to the asphalt mix remains constant throughout the experiment. As the number of cycles increases, the stress in the asphalt mix gradually de-

creases due to the decreasing stiffness of the material over time. This behavior is a result of the viscoelastic nature of asphalt, where prolonged loading leads to a reduction in its stiffness and subsequent decrease in stress.

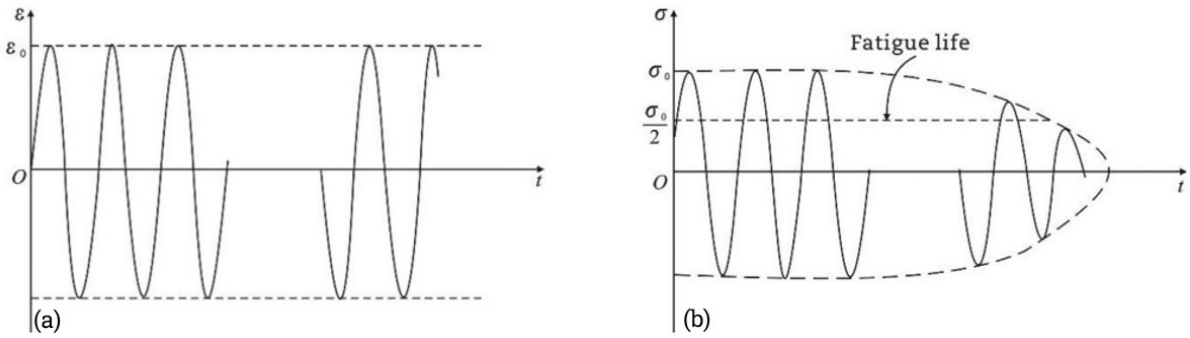


Figure 2.27: Constant strain test [54] (a) strain; (b)stress

The figure 2.28 reveals that different loading modes cause different fatigue test results.

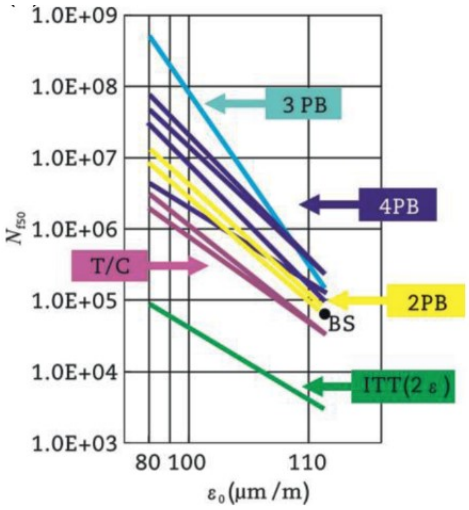


Figure 2.28: Different fatigue test results [8]

According to recent studies by Cheng et al. [125, 126, 127], Four Point Bending (4PB) testing has been proposed as a more accurate method for predicting the fatigue behavior of asphalt mixtures. Compared to other testing methods, such as UC/UT (uniaxial compression/tension) and IDT (Indirect Tensile test), 4PB testing is believed to provide results that are closer to simulating the actual behavior of asphalt layers in the field.

Monismith et al. has devised a novel approach to quantify the distinct fatigue characteristics exhibited under different loading modes, introducing the concept of the mode factor (MF) [63]. The equation employed to calculate the mode factor is as follows:

$$MF = \frac{|A| - |B|}{|A| + |B|} \tag{2.66}$$

Where A represents the rate at which tensile stress changes in the asphalt layer as its stiffness decreases by a specific percentage, while B represents the rate at which tensile strain changes in the asphalt layer as its stiffness declines by a specific percentage.

The mode factor (MF) is used to quantify the fatigue behavior of asphalt layers under different loading modes. In controlled strain mode, the MF is calculated as 1.0, while in controlled stress mode, the MF is -1.0. For asphalt layers with varying thicknesses and stiffness moduli, the MF approaches 1.0 for thin and soft layers, indicating a controlled strain mode, and approaches -1.0 for thicker and stiffer layers, suggesting a controlled stress mode. This can be seen from the figure 2.29. The MF provides insights into the dominant loading mode experienced by the asphalt layer during fatigue.

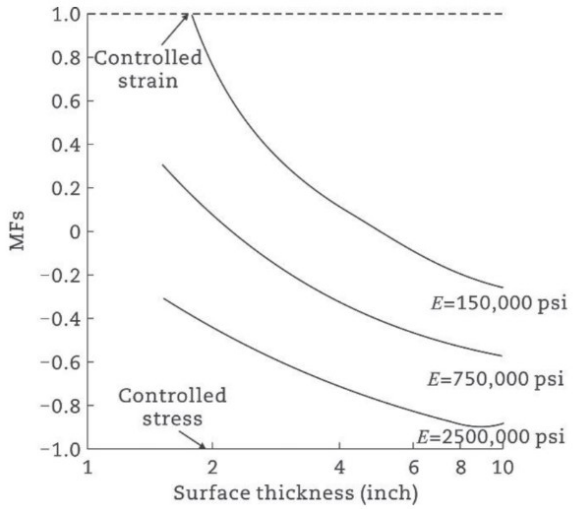


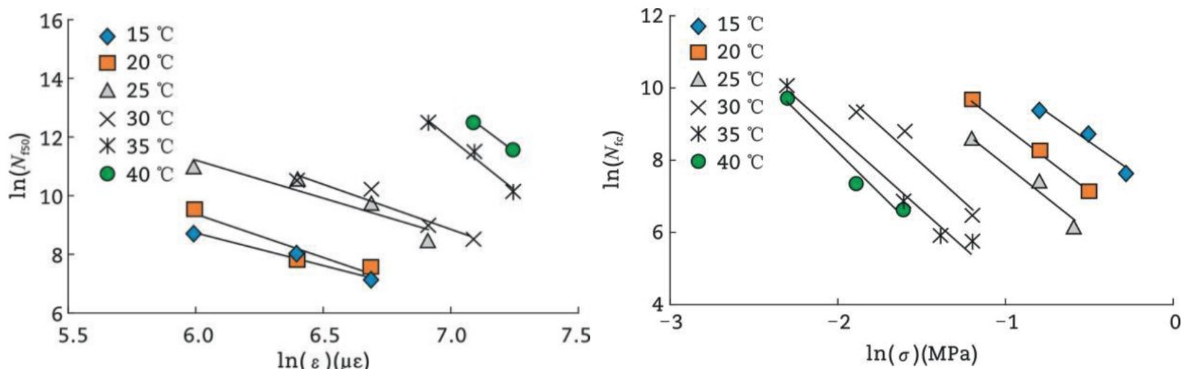
Figure 2.29: The MFs of asphalt layers with different thicknesses and stiffness moduli [128]

2.5.2. Effect of specimen size

Studies have shown that in both four-point bending and two-point fatigue tests, larger specimens tend to have a shorter fatigue lifespan compared to smaller specimens [129, 130]. Furthermore, larger specimens exhibit a higher rate of stiffness degradation compared to their smaller counterparts [54]. As a result, standardizing specimen size, along with other relevant parameters, becomes crucial in order to obtain reliable laboratory data and accurately assess the fatigue life of asphalt materials. This research deals with 4PB which uses standardized sizes and this effect is not studied and is not relevant.

2.5.3. Effect of temperature

Asphalt, being a viscoelastic material, is influenced by temperature due to the presence of binder bitumen. It has been proved that fatigue life of asphalt depends on temperature [64, 131, 132, 133, 134]. The rigidity of asphalt is reduced by a factor of three for every 10-degree Celsius increase in temperature [135, 136, 137, 54]. During fatigue tests, the slope obtained from the four-point bending test becomes steeper as the temperature rises, as observed by Bodin et al. (2010) [138] and shown in Figure 9 [54]. However, it is important to note that the internal heating effect on stiffness degradation is minimal compared to the damage caused [139, 54].



(a) 4 point bending **(b)** IDT

Figure 2.30: Effect of different temperatures on fatigue life[65]. (a) 4PB test. (b) IDT test

The fatigue life of asphalt is influenced not only by temperature but also by the combination of temperature and specific loading modes, as depicted in figure 2.30a [8]. The figure illustrates that an increase in temperature leads to an increase in fatigue life. However, in the case of the Indirect Tensile Test (IDT), there is an inverse relationship between temperature and fatigue life, as higher temperatures result in a decrease in fatigue life [8]. This highlights the significance of temperature effects on fatigue life and emphasizes the importance of selecting appropriate temperatures for fatigue testing.

2.5.4. Fatigue loading patterns

In the section 2.5.1, various testing methods for obtaining fatigue life data of asphalt mixtures have been discussed. This section provides a detailed analysis on the loading patterns and their relation to the behavior of field asphalt layers.

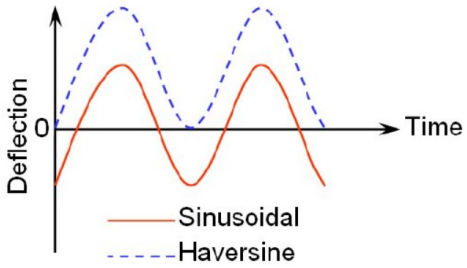


Figure 2.31: loading waveforms used for fatigue test [140]

In the figure 2.31, loading patterns such as sinus and haversine have been shown which are mainly used for fatigue tests. One of the most important concepts under loading patterns is the shape of the loading pulse. Figure 2.31 shows the most common shape of loading pulses applied in fatigue tests [8, 141, 118, 101, 94]. From the figure 2.31, it is clear that the haversine-wave test maintains continuous tensile strain, while the sinusoidal-wave test exhibits a tensile-compressive strain response. Type of loading waveform affects fatigue life and fatigue behavior with the sinusoidal wave generally resulting in a higher fatigue life than the haversine wave [8, 142, 94]. It should be noted that in loading pattern under vehicular load from the field asphalt layer is different than sin or haversine loading waveform. The strain waveform observed in the field asphalt layer differs from the commonly used haversine/sinusoidal wave, as it contains a compressive loading zone and exhibits strain superposition due to multi-axle loading. The haversine/sinusoidal wave fails to accurately simulate these characteristics. Recent studies by Cheng et al. [109, 143] have utilized actual strain waveforms measured from the field asphalt layer, including single-axle and tandem-axle waves, to assess the fatigue behaviors of asphalt mixtures [8].

Therefore, selecting a strain waveform that mimics the real field conditions is crucial for accurate fatigue testing of asphalt mixtures.

2.5.5. Effect of loading frequency

Loading frequency is important to know in order to understand fatigue behaviour and in order to propose fatigue monitoring models. Different researchers proposed loading frequency formulas like the equations (2.67) and (2.68).

$$f = \frac{\nu}{0.169de^{-0.00941\nu} + 0.036\alpha} \tag{2.67}$$

$$f = 0.0984e^{0.016T}\nu \tag{2.68}$$

Where ν is vehicular speed $\frac{km}{h}$, d is pavement depth, α is the radius of load (cm), T is the temperature ($^{\circ}$Celcius) and f is loading frequency.

Loading frequency is therefore crucial parameter for fatigue tests. According to standards frequency of 30 Hz is used. The impact of this parameter on mixture's fatigue life depends on the fatigue failure criterion that has been used for determining the fatigue life. If stiffness-modulus-based fatigue failure is used then the effect of loading frequency of fatigue life becomes minimum [8].

2.6. Machine learning approach

This research makes use of Artificial Intelligence (AI) in order to predict strain values based on voltage that comes from using the piezoelectric sensor as explained in detail in chapter "Methodology". AI is said as Machine Learning (ML) algorithms or ML-models that has been used is a subset of AI [144]. This type of model or methodology can be said to base the decision on the collected data [145] or in other words it is data-driven models [146]. More details on ML pipeline and its architecture can be found in chapter "Methodology" section 3.4. ML-algorithms that have been chosen and used in this research are supervised learning models which require user interaction. There are three types or subsets of machine learning [147]:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

Section 2.7 goes more in detail on each of the algorithms that have been used in this research. In other types of machine learning like reinforcement learning is what this current stage can be upgraded into for real-time structural health monitoring and hence this is one of the visions of this thesis. Unsupervised on the other hand learns patterns in the data without data needing to be labeled like clustering [147]. Supervised learning, unlike unsupervised learning, needs labelling to make predictions [148] which for example in this research are voltages from different frequencies. Predictions from supervised learning can be:

- Classification
- Regression

Classification type as its name suggests classify observations meaning categorizes observations in groups while regression type predicts or forecasts continuous values. This research deals with prediction problems hence regression models are used with more great detail provided in chapter three "Methodology".

2.7. Machine learning models or algorithms

In this section, an overview is given of the background and mathematical formulation of the regression algorithms that have been used in this research. The regression analysis has been

conducted within this research, as regression analysis represents a statistical technique employed to evaluate the relationships between dependent variables, also referred to as criteria, and one or more independent variables, known as predictors [149].

Machine learning tools or algorithms that have been used can be divided into groups such as Ensemble models, Artificial Neural Network models, Tree based models, Instance-based models, Support Vector regression, and Linear models. This has been also shown in figure 2.32. In this research, both machine learning and deep learning (Neural Networks) have been used which are subsets of Artificial Intelligence (AI)

For this research, the supervised learning route has been chosen with the aim that in the future it can be upgraded to unsupervised and even reinforcement learning. Supervised learning can be summarized as below [150]:

Training data comes in pair of inputs (x , y)

Where $x \epsilon R^d$

Training data can be denoted as:
$D = (x_1, y_1), ..., (x_n, y_n) \subseteq R^d \times C$

Where:
R^d is d-dimensional feature space
x_i is the input vector of the i^th sample
y_i is the label of the i^th sample
C is the label space

Data points (x_i, y_i) are drawn from distribution P(X,Y) which is unknown. It should be noted that the ultimate goal is to find function **h** so that a new pair $(x, y) \simeq P$

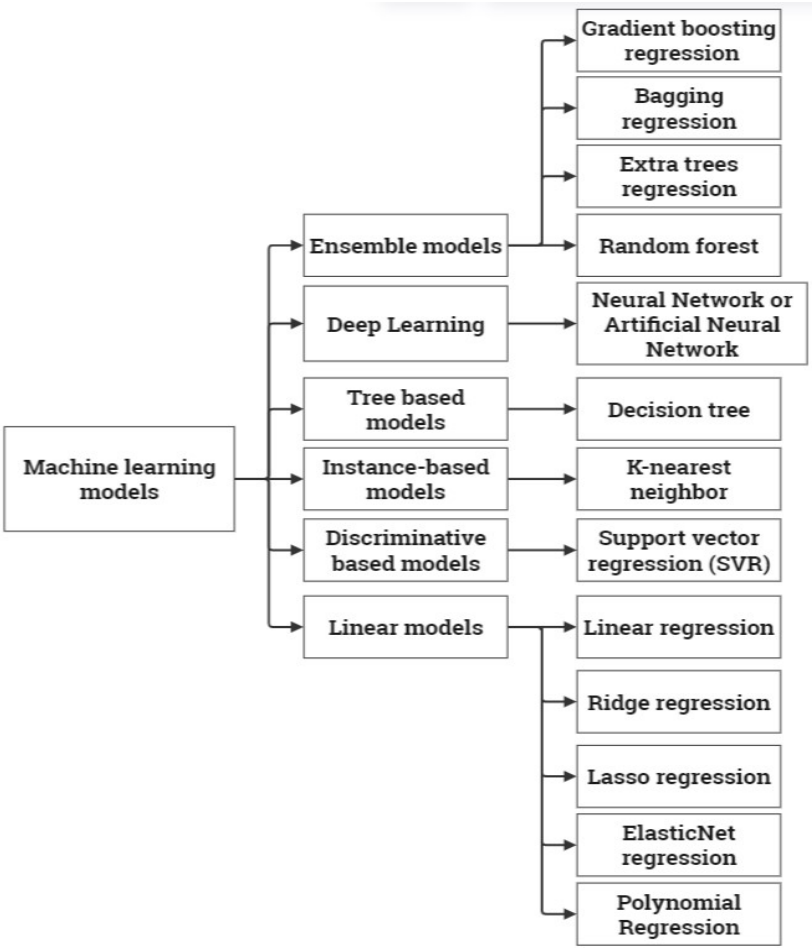


Figure 2.32: Machine learning algorithms used

2.7.1. Tree based models

A tree data structure serves as a hierarchical arrangement for the efficient organization and retrieval of data. Comprising interconnected nodes linked by edges, it exhibits a hierarchical arrangement among these nodes [151, 152, 153]. The uppermost node, known as the root, presides over a hierarchy of nodes termed child nodes. Each node possesses the potential for multiple child nodes, further expanding into recursive, self-replicating structures [151, 153]. This is illustrated in the figure 2.33.

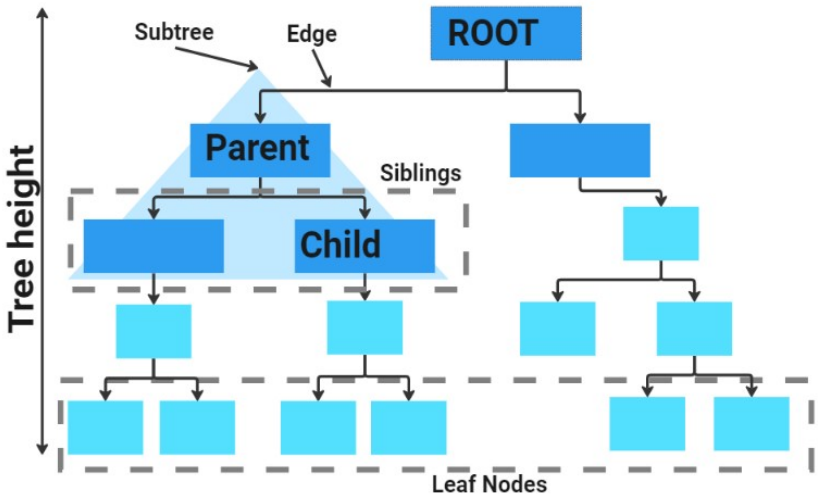


Figure 2.33: Tree data structure

Every constituent parts have been explained as follows as also explained in the references[151, 152, 153, 154]

Parent Node: The parent node of a given node is defined as the preceding node within the hierarchical structure.

Child Node: A child node, conversely, is the immediate successor of a specific node within the hierarchy.

Root Node: In a tree structure, the topmost node, known as the root, stands alone as the sole connection point to all other nodes in a non-empty tree.

Leaf Node: Nodes devoid of child nodes are designated as leaf nodes. This is also known as External Node.

Ancestor of a Node: Ancestors of a given node encompass all predecessor nodes along the path leading from the root to that specific node.

Descendant: Conversely, descendants pertain to all successor nodes along the path originating at a leaf node and extending to the node in question.

Sibling: Nodes that share a common parent node are colloquially referred to as siblings.

Level of a node: A node's level is ascertained by tallying the edges traversed from the root node to that node.

Internal node: An internal node is typified by its possession of at least one child node.

Neighbour of a Node: Neighbors of a given node comprise both its parent and child nodes.

Subtree: A subtree is delineated as any node within the tree, inclusive of all of its descendant nodes.

There are three different types of trees as shown in figure 2.34. Various types of tree data structures are used for organizing and storing data efficiently. Each of these tree types can be even further subdivided in different groups and types but that is not covered here.

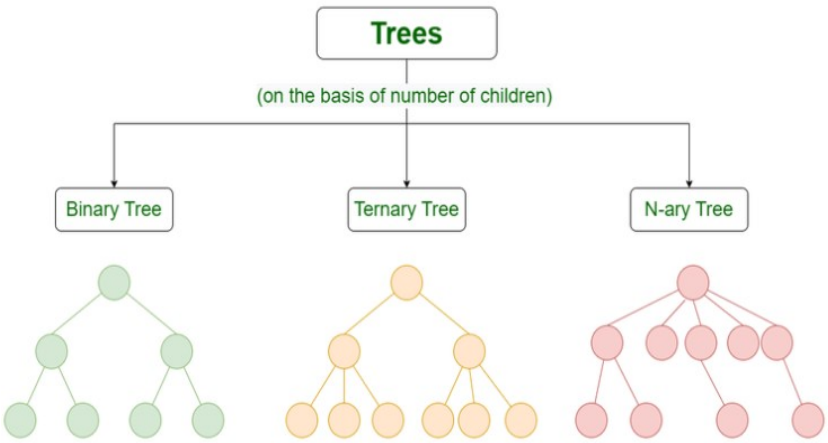


Figure 2.34: Tree data structure types [151]

Binary Tree: A binary tree is a hierarchical data structure wherein each node is constrained to possess a maximum of two interconnected children [152]. There are different variations of binary trees such as full binary trees, complete binary trees, balanced binary trees, and also degenerate or pathological binary trees [152].

Ternary Tree: The ternary tree represents an additional tree data structure, wherein every node has the capacity to accommodate a maximum of three child nodes, conventionally identified as "left," "mid," and "right." This structure surpasses the binary tree paradigm by offering three distinct branching options at each node[152].

N-ary Tree or Generic Tree: N-ary trees, often referred to as generic trees, are versatile data structures consisting of nodes. Each node in an N-ary tree contains records and a list of references to its children. Unlike linked lists, N-ary trees can store multiple references within each node, enabling efficient representation of hierarchical data where nodes may have varying numbers of children.

These tree structures serve as foundational tools in various computer science applications, allowing for organized and efficient data storage, retrieval, and manipulation. This is the reason that they are also used in ensemble models such as Random Forest, Extra Trees regression, etc.

2.7.1.1. Decision tree

Decision trees are also used in Ensemble models. A decision tree is a supervised machine learning algorithm that operates by constructing a flowchart-like structure to facilitate decision-making and predictions [153] and can be seen as nested if statements [153]. This structure, resembling a tree, is created through a sequence of conditional control statements that interrogate the data's attributes and generate new branches accordingly. It begins with a single decision node and diverges into two or more nodes, which can either be additional decision nodes or prediction nodes, depending on the data's characteristics. Decision trees are versatile, as they can be employed for both classification [155] and regression tasks. A decision tree can be of many types as shown in figure 2.34 however, a simple Decision Tree structure may look like as shown in figure 2.35. For maths and a detailed explanation of the algorithm, chapter 9 of the book "DATA MINING AND KNOWLEDGE DISCOVERY HANDBOOK" is referred to [154]. The pseudocode used for constructing a decision tree can be found in the source referred to as [156].

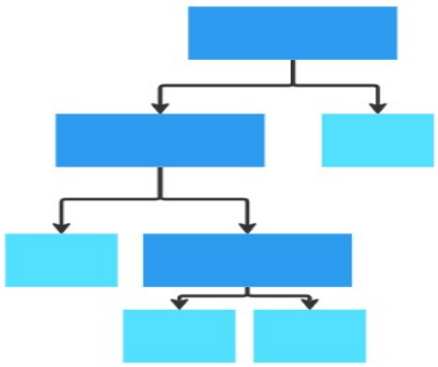


Figure 2.35: Decision tree structure

2.7.2. Ensemble models

One of the cutting-edge solutions is known as ensemble methods, which enhance predictive performance by utilizing or combining multiple inducers to arrive at a better decision and prediction [157, 145]. An example of an ensemble inducer can be any type of machine learning algorithm such as decision trees, linear regression, etc.

Decision trees are nothing more than estimators that are capable of learning simple decision rules [158]. More details about tree structure and types are given in the section Tree-Based Models and Decision Tree. If models are not too complex. As complexity increases, such trees will also start to increase leading to less accurate predictions and increases in rigidness making it less possible to add new data [159]. However, there are techniques where decision trees can be combined, split, and used in order to predict or classify something with an increase in efficiency and accuracy.

2.7.2.1. Gradient boosting regression

Gradient boosting is a versatile machine learning algorithm developed by Robert Schapire in 1990 for tabular data, capable of handling complex relationships, missing values, outliers, and categorical features [160]. Understanding its workings becomes essential when fine-tuning hyperparameters and customizing loss functions for improved model performance [160]. Gradient boosting is an ensemble method involving multiple weak models to enhance overall performance [160, 161] which was first thought by Kearns and Valiant in 1988 [162]. Gradient boosting builds an additive model progressively, optimizing the loss function, and at each step, fits a regression tree to the negative gradient of the loss function [163]. The colors in the figure 2.36 are already explained in section 2.7.1

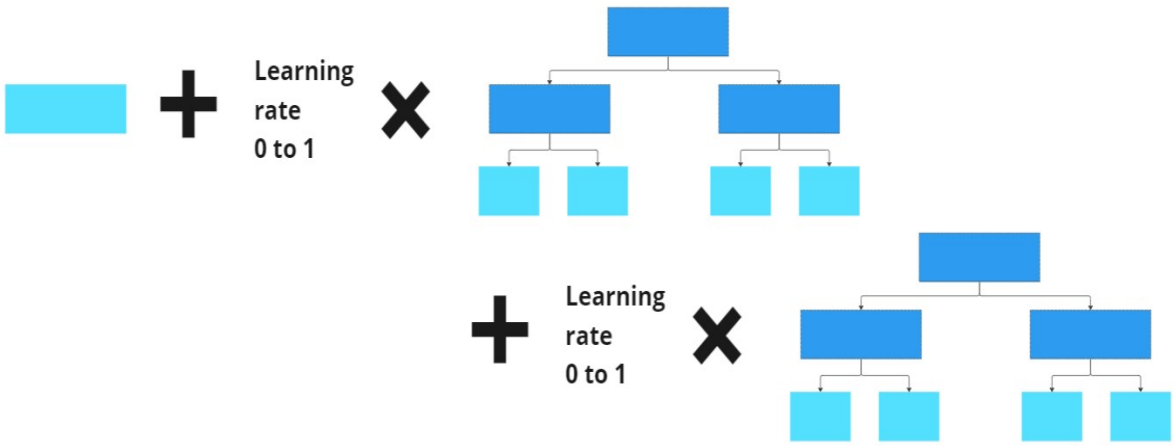


Figure 2.36: Gradient boosting regression

In gradient boosting, the initial base tree is constructed based on the average of the target values, as mentioned in [164]. This average value can be thought of as the value assigned to the first leaf node in the tree. In the context of gradient boosting, the residuals derived from the initial prediction in relation to the target variable serve as the new prediction target. This iterative process sequentially constructs decision trees with the aim of reducing the errors from the preceding tree [145]. The ultimate prediction is obtained by summing all the predictions and taking their average. However, this summation can potentially lead to over-fitting.

Selecting an appropriate number of shallow trees in a GBM model, as opposed to fewer but deeper trees in a random forest, is crucial for preventing over-fitting or under-fitting during training [160]. To mitigate this, the predicted residuals are incrementally scaled to gradually adjust the averaged value [165]. In Figure 2.36, the scaling, also referred to as the learning rate is applied to the tree which is between zero and one. This ensures the problem with over-fitting and small steps which leads to smaller pseudo-residuals can ultimately lead to the better prediction values.

Mathematical formulation is as following [166]:

$$\hat{y}_i = F_M(x_i) = \sum_{m=1}^{M} h_m(x_i) \tag{2.69}$$

Where: $\hat{y}$ is predicted value for x_i
h_m is fixed size decision trees
M is the number of estimators

$$F_m(x) = F_{m-1}(x) + h_m(x) \tag{2.70}$$

After applying first order taylor approximation to the loss function $l(y_i, F(x_i))$ and removing the constant terms, we get [167]:

$$h_m \approx \arg\min_h \sum_{i=1}^{n} h(x_i) g_i \tag{2.71}$$

2.7.2.2. Bagging regression

Bagging also known as bootstrap aggregation and boosting are fundamental ensemble learning techniques [168]. Bagging concurrently trains weak learners while boosting trains them sequentially, progressively amplifying the importance of misclassified data to refine model parameters [168].

Bagging, a method introduced by Leo Breiman in 1996, involves training multiple independent models by randomly sampling instances with replacement from the original dataset [160]. These models are combined through majority voting (in case of classification) or arithmetic averaging of decision tree outputs [169]. Variations include Improved Bagging Algorithm (IBA) and other techniques like Online Bagging and Wagging, which assign random weights to instances[160]. Bagging has also been effective in facial recognition and can be seen as a precursor to the Random Forest technique [170].

Bagging has been summarized below [171]:

1) Sample m data sets $D_1, ..., D_m$
2) For each D_j train classifier $h_j()$
3) Final classifier is $h(x) = \frac{1}{m} \sum_{j=1}^{m} h_j(x)$

2.7.2.3. Extra trees regression

Extra tree regressor [172, 173] also known as Extremely Randomized Trees [174] which is built as an extension of random forest [173] gives less variance and have more accuracy than any generalized model [175] and is less likely to over fit a dataset [173]. It is also from sklearn.ensemble module and it is an averaging algorithm based on randomized decision trees [172]. It has been observed earlier that going from a decision tree to a single extra tree increases bias however this is limited to a certain depth as using multiple extra trees can vastly improve prediction accuracy as also observed in the results of this research. The structure of Extra Trees (classification) is summarized in figure 2.37 [176, 177]. It is essential to highlight that in classification tasks, predictions rely on majority voting among decision trees, whereas in regression, as used in this study, predictions are derived by averaging the outputs of decision trees [174].

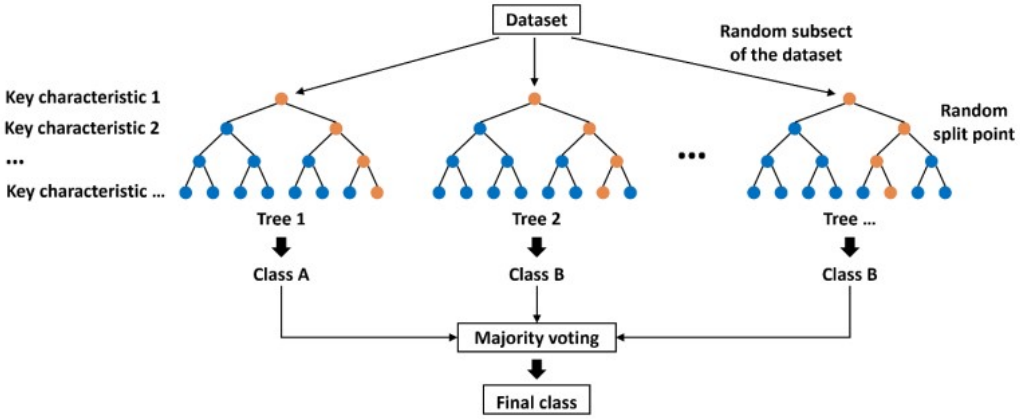


Figure 2.37: Structure of Extra Trees [176]

Extra Trees is fundamentally similar to Random Forest which also combines predictions from many decision trees and it can be seen as having better performance than Random Forest algorithm [174] which is also observed in the results of this research. The Extra-Trees algorithm, distinct from other tree-based ensemble methods, constructs an ensemble of unpruned decision or regression trees employing a top-down approach [174]. It distinguishes itself from other ensemble methods through the utilization of entirely random cut-points for node splitting and the incorporation of the entire learning sample during tree growth, in contrast to bootstrap replicas used by traditional methods [174].

2.7.2.4. Random Forest

Among the renowned bagging algorithms, the Random Forest stands out as it is widely applicable to a range of classification and regression predictive modeling problems [178]. The term "random forests" originates from the utilization of multiple decision trees to predict [159]. Essentially, a Random Forest consists of bagged decision trees as it is an extension of bootstrap aggregation (bagging) [178], incorporating a slightly adapted splitting criterion [171]. The model construction process begins with the creation of a bootstrapped dataset [179], involving a random selection of data points from the original dataset, potentially with repetitions. Subsequently, trees are constructed by randomly choosing subsets of features from this bootstrapped dataset and repeating this procedure iteratively. This approach introduces diversity to the model as various combinations are incorporated into the trees. The ultimate prediction is obtained by averaging the target values predicted by all the individual trees.

Unlike extra trees, the random forest gives medium variance [175] and it is also an averaging algorithm based on randomized decision trees [172]. Both Extra Trees and Random Forest are composed of a large number of decision trees which gives results or predicts result taking into account every tree [176]. The difference between Random Forest to Extra Trees is selection of cut points in order to split nodes [176] as can also be seen from figure 2.37. Random Forest does this by choosing the optimum split which results in more computation time while Extra trees do it

randomly making the algorithm to be execute much faster [180] as no time needs to be taken for finding any optimum split.

In contrast to bagging, the random forest method introduces an additional step by selecting a subset of input features (columns or variables) to consider at each split point during tree construction [178]. Random Forest generates numerous trees, with each tree being exposed to a partial subset of the training dataset, subsequently making predictions based on this subset. The final prediction is obtained by aggregating the predictions from all the trees [177]. Figure 2.38 is similar to figure 2.37 but as explained in this section the cut points are different

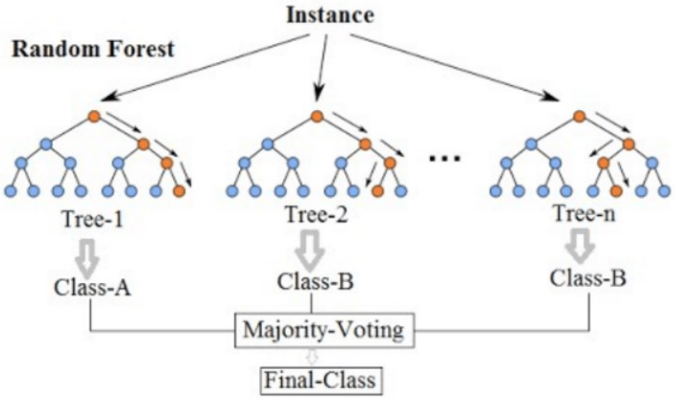


Figure 2.38: Structure of Random Forest [177]

2.7.3. Artificial Neural Networks

Artificial Neural Networks (ANNs) Neural Networks (NN) or Simulated Neural Networks (SNNs) are names of the same type of algorithm or part of the same machine learning subset [181]. These are the backbone of deep learning algorithms and are inspired by human brain neurons hence they are called neural as they also mimic how neurons in the brain work [181]. In figure 2.39 we see the structure of a neural network if it consists of 3 neurons. In this research (500,500) neurons have been used in the neural network model.

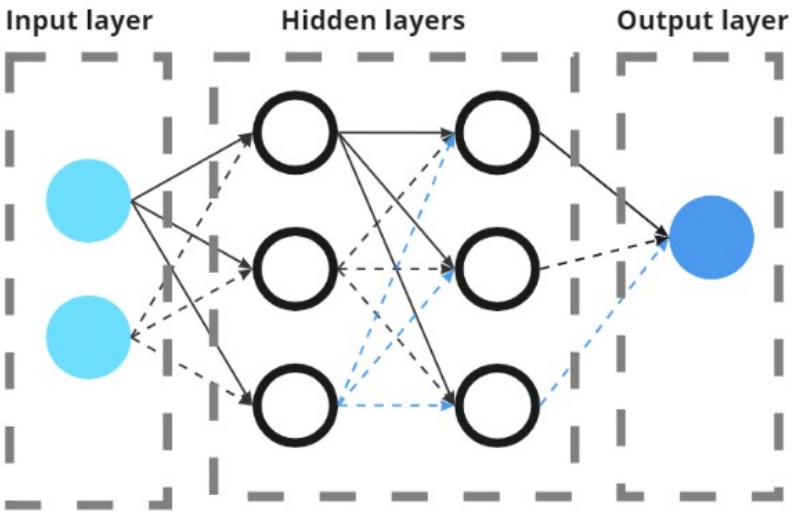


Figure 2.39: Structure of Neural Network

Neural networks consist of interconnected layers of nodes, encompassing input, hidden, and output layers as shown in figure 2.39. The input layer directly receives the data which is then

went through hidden layers where intermediate computation takes place and then the output layer creates the required output [182]. These artificial neurons possess individual weights and thresholds. Activation occurs when a neuron's output surpasses its threshold, forwarding data to the subsequent layer; otherwise, no data transmission occurs [181]. This whole process of the neural network begins with perceptron or the idea of perceptron which in simple terms means that the perceptron receives inputs, multiplies them by some weight, and then passes them to activation function (such as logistic, relu, tanh, etc) to produce results as outputs [182].

Training neural networks with data refine their accuracy. Once fine-tuned, these networks serve as potent tools in computer science and AI, enabling rapid data classification and clustering. Neural networks significantly expedite tasks like speech and image recognition, reducing manual processing times from hours to minutes. A notable illustration of a neural network application is Google's search algorithm [181]. It should be noted that if a neural network model consists of more than three layers, it is then considered a Deep Learning (DL) algorithm [181]. In this research ANN has been used however there are also other types of neural networks such as Convolution Neural Networks (CNN), Recurrent Neural Networks (RNN), etc. The detailed mathematical formulation can be found in scikit-learn documentation [183] and in a short article [184, 185].

2.7.4. Instance based models

Instance-based learning, frequently denoted as memory-based learning or lazy learning, is a computational paradigm that relies on the retention of training data to formulate predictions for forthcoming data points [186] based on some similarity measure [187]. Unlike some methods, it doesn't necessitate prior data assumptions or knowledge, rendering it straightforward to implement and grasp. Nonetheless, it can be computationally demanding as it requires storing the entire training dataset in memory before making predictions [186] but it can be faster for less data as its time complexity depends on the size of training data [187]. Furthermore, this approach doesn't exhibit strong generalization to unseen datasets, as it relies on memorized instances rather than learned models for predictions [186].

Some of examples of this type of model are:

1. K Nearest Neighbor (KNN)
2. Case-Based Reasoning
3. Learning Vector Quantization (LVQ)
4. Locally Weighted Learning (LWL)
5. Self-Organizing Map (SOM)

Another type of machine learning methodology is model-based learning which demands more effort and gives much better generalization that can be used alongside with instance-based machine learning method [186]. However, this research only deals with instance-based models like K Nearest Neighbor alone alongside using other machine learning models.

2.7.4.1. K-nearest neighbor

K-nearest neighbor is an instance-based model which is very popular as it can be used for both classification and regression problems [188]. In the training phase, K-Nearest Neighbors (KNN) stores the entire dataset and computes distances to predict based on a chosen metric like Euclidean distance [188] or cosine similarity [186]. In the context of regression tasks, this approach involves the computation of either the arithmetic mean or a weighted mean of the target values associated with the K nearest neighbors. This computed mean serves as the basis for predicting the value of the input data point.

KNN which is non-parametric [186] is a widely used, intuitive algorithm, but its effectiveness depends on choosing the right K and distance metric, requiring meticulous parameter tuning for best outcomes. For detailed algorithm and math, this short article is referred [189]

Pseudo code for K-nearest neighbor is explained in **Algorithm 1**.

Algorithm 1 KNN Algorithm [188]

Load data, initialize k
for each training point **do**
  Calculate distance to test point
  Get top k distances
  Predict class with majority vote
  Return predicted class
end for

2.7.5. Discriminative based models

Discriminative models, a category within machine learning, operate by directly mapping input data to their respective classes, eschewing the explicit modeling of underlying probability distributions or intrinsic data structures. In contrast to generative models, which seek to encapsulate the data generation process, discriminative models are primarily focused on classification or regression tasks. Their primary objective is to acquire the decision boundary or function that most effectively segregates distinct classes or predicts target values based on input features.

Within this realm, Support Vector Machines (SVMs) and Decision Trees (DTs) are noteworthy examples of discriminative models. SVMs, in particular, are recognized as maximal margin classifiers, denoting their ability to learn decision boundaries that optimize the spatial separation between samples belonging to different classes. Often, SVMs employ kernel functions to facilitate this process. Notably, this separation distance can be leveraged to impart a degree of softness to the classifier, enhancing its adaptability when dealing with data points in proximity to the decision boundary. Conversely, Decision Trees gain insights into the decision boundary through a recursive partitioning procedure, with a focus on maximizing information gain or other defined criteria. This process culminates in the delineation of distinct regions within the feature space, each corresponding to a specific class.

2.7.5.1. Support Vector regression (SVR)

Support vector regression (SVR) is a supervised machine learning technique for regression tasks [190, 191]. Qingyun Ge et al. provide pseudocode for SVR [192]. The Support Vector Machine (SVM) is a discriminative algorithm designed to seek the optimal hyperplane for clear demarcation of data points within an N-dimensional space (N being the number of features) [193]. In the context of a two-dimensional space, this hyperplane manifests as an optimal line that effectively separates data points into two distinct classes [193]. However, in higher-dimensional spaces, the hyperplane takes on a more complex form beyond a mere line. Consequently, SVR's optimization process is characterized by its reliance on support vectors, which constitute a limited subset of training data samples [194]. Remarkably, the optimization solution remains unaffected by the input data's dimensionality, being solely contingent on the count of support vectors [194]. Support Vector Regression (SVR) leverages kernel functions to efficiently address nonlinear regression problems. This is achieved by mapping the original feature space into a higher-dimensional kernel space, where the data can be effectively separated using linear methods, thus enabling the resolution of complex nonlinear regression challenges [194]. More details on its math and algorithm can be found in chapter 7 "Support vector regression" written by Fan Zhang, Lauren J. O'Donnell [194] and article written by Sharif [193] and by MathWorks [195].

2.7.6. Linear models

2.7.6.1. Linear regression

Regression analysis is a statistical methodology employed for the purpose of investigating the relationships that exist between dependent variables, often referred to as criteria, and one or more independent variables, commonly recognized as predictors [196, 197]. The equation (2.72) is taken from [196].

$$y = a_1x_1 + a_2x_2 + a_3x_3 + ... + a_nx_n + b \tag{2.72}$$

Where:

- y is the target variable.
- $x_1, x_2, x_3, ..., x_n$ are the features.
- $a_1, a_2, a_3, ..., a_n$ are the coefficients.
- b is the parameter of the model.

The parameters a and b of the model are selected through the Ordinary least squares (OLS) method. It works by minimizing the sum of squares of residuals (actual value - predicted value) [196].

2.7.6.2. Ridge regression

Ridge regression represents an extension of linear regression [198] and takes even one more step than Lasso regression [199], wherein the loss function undergoes modification aimed at reducing model complexity [196]. This adaptation entails the introduction of a penalty parameter equal to the square of the coefficient magnitudes. Ridge regression tries to fit by minimizing residual sum of squares (RSS) or cost function [198]. This not only makes the weights to have smaller absolute values but also penalizes extremes resulting in evenly distributed weights [199].

Loss function = OLS + alpha * summation (squared coefficient values)

Where OLS is Ordinary Least Squares. Example of cost function is [198]:

$$\frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - h(x^{(i)}))^2 \tag{2.73}$$

Where $h(x^{(i)})$ is hypothetical function
$y^{(i)}$ is value of target variable for i^{th} example.

In the context of the loss function, the parameter alpha assumes a critical role, necessitating careful selection. A low alpha value has the potential to induce over-fitting, whereas a high alpha value may result in under-fitting [196].

2.7.6.3. Lasso regression

Lasso regression, which stands for Least Absolute Shrinkage and Selection Operator, constitutes a modification of linear regression [196]. In Lasso regression, the adjustment to the loss function aims to minimize model complexity by limiting the sum of absolute coefficients, also known as the l1-norm [196, 200, 199].

The loss function for Lasso Regression can be expressed as in equation (2.74) [196].

$$L_f = OLS + \alpha * \sum(|C_o|) \tag{2.74}$$

In the loss function (2.74), the parameter alpha serves as the penalty parameter that requires selection or optimization. [196], OLS is Ordinary Least Squares and C_o is the magnitude of the coefficients.

2.7.6.4. Elastic-Net regression

ElasticNet combines Ridge and Lasso regression properties [196, 199]. It penalizes using both l1 and l2 norms, controlled by the $l1\_ratio$ coefficient [196].

$$L_f = OLS + r \cdot \alpha \cdot \sum|C_o| + \frac{1-r}{2} \cdot \alpha \cdot \sum({C_o}^2) \tag{2.75}$$

Where L_f is the lost function, OLS is Ordinary Least Squares, C_o is the magnitude of the coefficients and r is another coefficient so when r = 0, it becomes similar to Ridge regression, and when r = 1, it becomes similar to Lasso regression.

2.7.6.5. Polynomial regression

Polynomial Regression is a variant of linear regression or an extension of linear regression where the association between the independent variable, denoted as x, and the dependent variable, represented as y, is expressed as a polynomial function of degree 'n' [201]. This method explores the nonlinear relationship between the variable x and the conditional mean of y, represented as E(y | x) [201]. Polynomial regression is a versatile technique applicable in scenarios where the underlying relationship between variables is believed to be polynomial in nature or when dealing with complex relationships where a polynomial serves as a suitable approximation [202].

Polynomial regression for a single predictor looks like:

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + ... + \beta_h X^h + \epsilon \tag{2.76}$$

In polynomial regression, the parameter "h" denotes the degree of the polynomial used in the model. Notably, each degree of the polynomial has a designated name; for instance, "h = 2" corresponds to a quadratic relationship, "h = 3" signifies a cubic relationship, "h = 4" represents a quadratic relationship, and so forth. Despite its ability to capture nonlinear associations between the dependent variable "Y" and the independent variable "X," it's important to note that polynomial regression is categorized as a form of linear regression [203]. This classification stems from the fact that it maintains linearity in terms of the regression coefficients $\beta_1, \beta_2, ..., \beta_h$, even while accommodating nonlinear data relationships.

To estimate the equation as described in Equation (2.76), typically, only the response variable (Y) and the predictor variable (X) are needed. However, polynomial regression models have the potential to incorporate additional predictor variables, which may introduce interaction terms. Thus, while the fundamental equation for a polynomial regression model is relatively straightforward, it's important to recognize that the complexity of the model can increase depending on the specific context and the inclusion of additional predictor variables. More details on its math and algorithm can be found in articles [203].

2.8. Conclusion

In this chapter, we provided a comprehensive overview of the pertinent literature and essential concepts related to the detection of damage and the prediction of fatigue life in Structural Health Monitoring (SHM) through supervised machine learning. The conclusions drawn in this chapter are instrumental in shaping the research direction. It was determined that the four-point bending test is the most suitable method for characterizing fatigue parameters in asphalt, and thus, we adopted this approach for our study. Additionally, this chapter offered a detailed examination of various fatigue life models and damage models. Moreover, different sensor types were analyzed, with a more focused exploration of piezoelectric sensors, especially the PZT sensor. For this study, it was decided to utilize a soft PZT attached to the substrate for the four-point bending test, allowing to exploration of a broader bandwidth encompassing varying strain and frequency levels. This approach, combined with the methodology employing supervised machine learning, represents a novel and innovative contribution besides providing comprehensive knowledge on different fatigue life and damage models developed for asphalt concrete in the past.

Furthermore, we introduced the concept of machine learning (ML) and presented 13 distinct ML algorithms employed in this study. These algorithms were supplemented with references to their pseudocodes and the associated hyperparameters, which were fine-tuned automatically.

The chapter also clarified that the choice to employ machine learning was rooted in the inherent complexity of the problem. ML streamlines the problem-solving process by eliminating the

need to derive equations for PZT sensor behavior in bending mode. Instead four-point bending tests were conducted on Teflon beams to generate extensive sensor data, encompassing voltage amplitude and its corresponding frequency. These data were subsequently used to train ML models. Another compelling rationale for the adoption of ML is its adaptability to various conditions, such as changes in strain levels and frequencies. Asphalt concrete exhibits distinct responses under varying environmental conditions, including alterations in temperature and frequency.

3

Methodology

3.1. Introduction

In this section, methodology is explained. Starting from section 3.1.1 it explains how materials are prepared, how data are collected what type of tests were done, and what instruments were required in order to gather the required data doing those tests/experiments. After this in section 3.2 general methodology is given for this research explaining on how to collect data for the training of machine learning models and how to use those models for new data to get strain, loading spectra and predict fatigue life.

In section 3.3 COMSOL finite element program is introduced and explained how it is used to verify results from Teflon experiments. In section 3.4 Machine learning pipeline is explained in great detail. machine learning pipeline starting from data pre-processing to optimization and cross-validation. In section 2.7 Machine learning (ML) models is further explained in detail. the models or different ML algorithms that have been used. In section 3.5 fatigue life prediction or monitoring models have been explained. It should be noted that ML training and use is also part of the structural health monitoring process.

3.1.1. Asphalt

An asphalt mixture known as Asphalt Concrete (AC) 16 Bin/Base 35/50 was manufactured at the Laboratorium Ontwikkeling Wegenbouw (Dura Vermeer) in Eemnes, Netherlands. This specific mixture comprised a minimum of 65% Reclaimed Asphalt Pavement (RAP) and was formulated with a target binder content of 4.3% by mass of binder, utilizing a 20mm penetration grade binder. The production process involved a temperature of 165 - 170 °C.

All the constituent materials utilized in this mixture were sourced and produced from the adjacent asphalt plant located at the Eemnes laboratory site. Detailed information concerning the relevant properties of this asphalt mixture can be found in the subsequent tables provided in this section.

Table 3.1: Sieve Analysis Data

Sieve size [mm]	%Passing		
	min	goal	max
31.5	100	100	100
22.4	99	100	100
16	94	97	100
11.2	75	81	85
8	62	68	72
5.6	53	58	63
2	41	43	45
0.5	15	29.3	35
0.125		9.1	
0.063	4	7	9

Table 3.2: Construction materials used for the mixture

Material Code	Name	Result of Matrix Calculation % (m/m)
4630 (v1.2)	Bestone 11/16 Norway	10.83
4620 (v1.2)	Bestone 8/11 Norway	4.67
4600 (v1.2)	Bestone 2/5 Norway	2.20
4610 (v1.3)	Bestone 4/8 Norway	4.15
5120 (v1.2)	Coarse Sand 0/2 Netherlands	10.97
6110 (v1.0)	Wigro 50K 0/0.1 Winterswijk	0.01
6190 (v1.0)	Own Material 0/0.1 AMI	0.80
7040 (v1.4)	Penetration Bitumen 160/220	1.37
	Various	
8023TT2207 (v1.2)	Milling Asphalt Subbase 0/16 APE	65.00

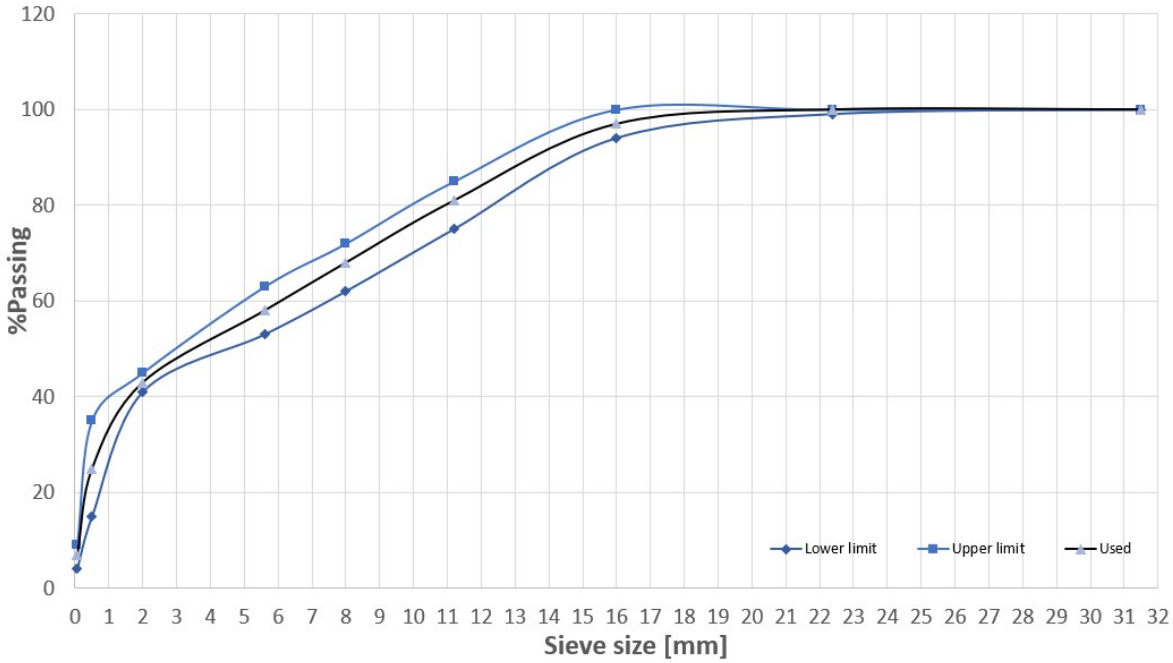


Figure 3.1: Percentage passing curve

A total of **26** asphalt beams, each measuring 400 by 50 by 50 units, were fabricated for the pur-

Table 3.3: Physical properties of binder used in the mixture

Properties	Technical specification	Value	Units
Penetration	NEN-EN 1426:2015	19	0.2-mm
Ring & Ball Temperature	NEN-EN 1427:2015	61.5	°C
Binder Percentage	NEN-EN 13108-1:2006	4.5	% m/m

pose of generating an SN curve. Additionally, **8** beams with identical dimensions were created specifically for utilization with sensors. Further pertinent properties are:

Table 3.4: Details of Mixture Parameters

Mixture characteristics	Values	Units
New bitumen type	160/220	
Pen mixture / Mixing temperature	35/50	165 °C
Bitumen % ”on”	4.5	% m/m op
Bitumen % ”in”	4.3	% m/m in
Target density	2390	kg/m3
Void Space	3.9	%
Square beam dimensions	50*50	mm

Table 3.5: Gyrator details

Parameter	Value	Units
Target density (ρ_{str})	2390	kg/m^3
Gyrator mold diameter	100	mm
Desired height	77	mm
Number of gyrations	100	st
Compaction correction gyrator	98.5	%
Compaction correction plate compactor	98	%
Plate thickness	90	mm
Amount of mixture to be produced	1479	g
Amount of asphalt to be compacted	1422	g
Bitumen % (’on’)	4.5	%
Bitumen % (’in’)	4.3	%
Mass of mineral for weighing record	1415	g

Table 3.6: Density Specifications

Density	Value	Units
Target density (ρ_{str})	2390	kg/m^3
Minimum density of the test specimen	2360	kg/m^3
Maximum density of the test specimen	2420	kg/m^3

Table 3.7: Basic plate dimension for fabrication

Parameter	Value	Units
Dimensions of mold		
Length	500	mm
Width	500	mm
Height	90	mm
Volume	22500	cm^3
Target density (ρ_{str})	2390	kg/m^3
Desired plate density as a % of target density	98	% (V/V)
Desired plate density	2342	kg/m^3
Bitumen content	4.3	% (m/m) "in"
Mixing temperature	165	°C
Amount of asphalt to be produced	54807	g
Amount of mineral to be weighed	52545	g
Amount of asphalt to be compacted	52700	g

Table 3.8: Mixture dry and wet mass

Parameter	Average values	Units
Water temperature	16	°C
Dry mass	2672	g
Submerged mass	1568	g
Wet mass	2675	g
Density of the test specimen	2413	kg/m^3

Prior to the gluing or testing phases, it is imperative that the test specimens undergo a meticulous drying process in a controlled environment, characterized by a relative air humidity of less than 80% and a temperature not exceeding 20 °C according to NEN-EN12697-24 as also summarized in the figure 3.2. To ensure precision in measurements, each test specimen should be subjected to a weighing interval of at least 4 hours. Should the difference in weight between consecutive weightings be found to be less than 0.1%, this indicates the attainment of a thoroughly dried state. Moreover, maintaining uniformity in the bulk density of individual specimens within a batch stands as a vital requirement. The permissible variation in bulk density among these specimens should not surpass 1% relative to the overall average density of the batch, signifying a commitment to consistent and accurate analysis.

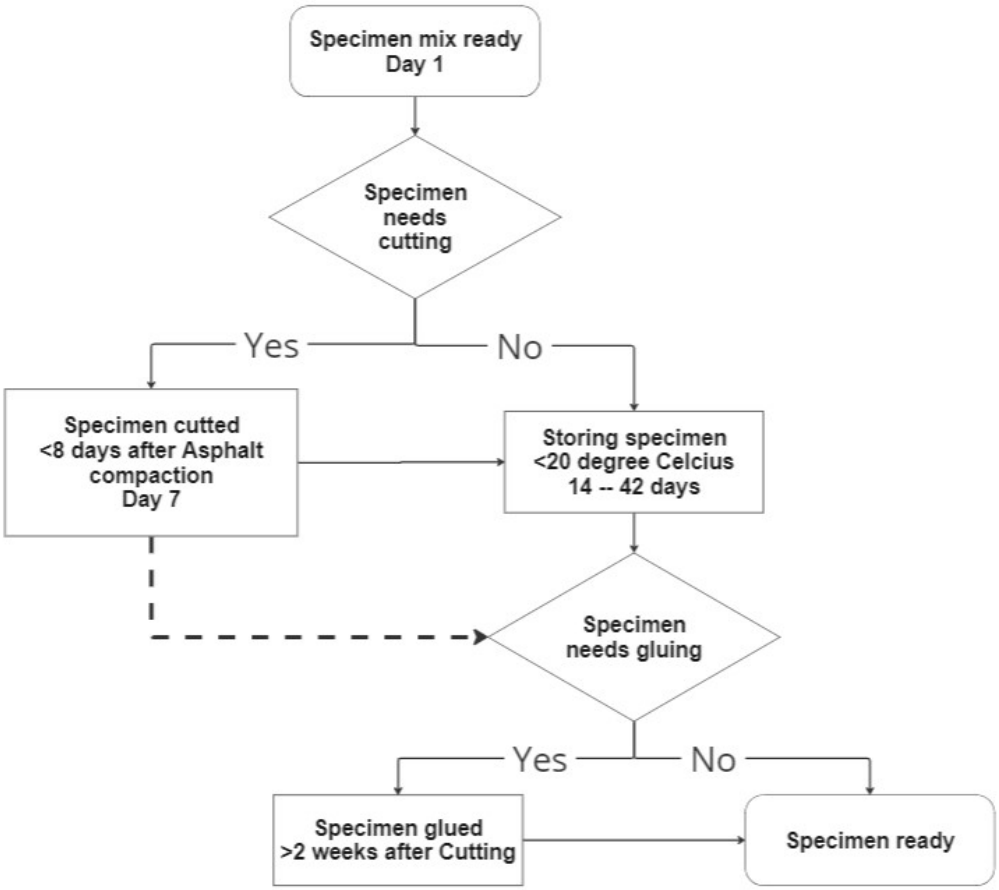


Figure 3.2: Flowchart for storing specimen according to NEN-EN12697-24

The images presented in figure 3.3 depict the finalized beams following the processes of fabrication, storage, and cutting.

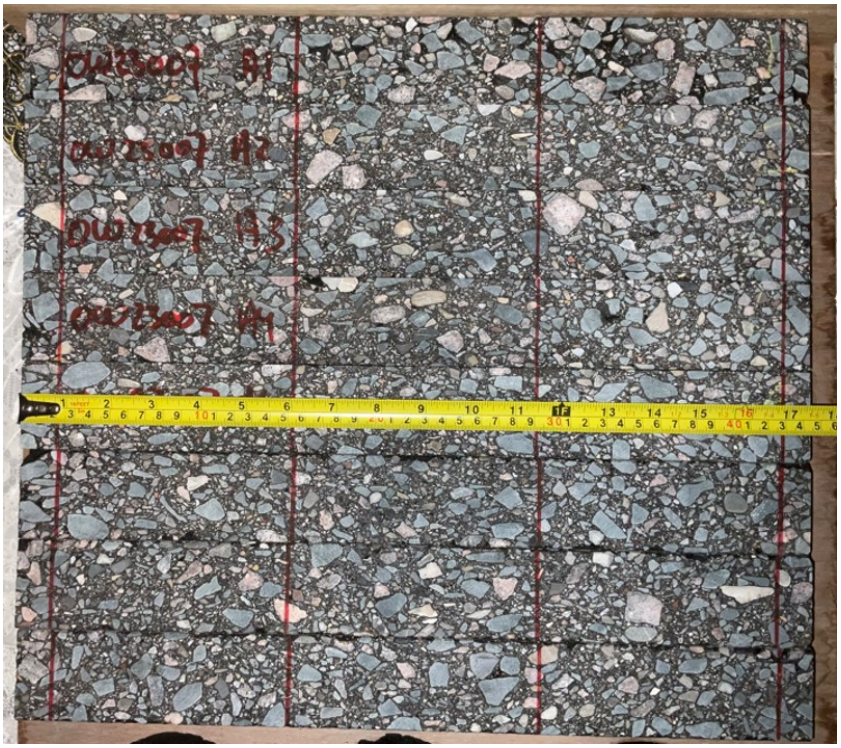


Figure 3.3: Asphalt beams

3.1.2. Piezoelectric Sensor

In the execution of the methodology, the seamless integration of the piezoelectric sensor into the experimental protocol was realized. The sensor was carefully attached underneath the substrate using strong double-sided adhesive, creating a secure and trustworthy connection that was crucial for the sensor to closely follow the substrate's movements. This arrangement enabled smooth interaction with an oscilloscope, which diligently recorded and saved the sensor's output signal onto a USB storage device. To enhance stability, the sensor was additionally fastened with duct tape, and special care was taken to ensure accurate electrode and electrical contact connections. Importantly, both ends of the oscilloscope were insulated, ensuring that the signal's reliability remained intact throughout the experiment. Duct tape on top of the sensor which is attached with strong double sided tape on the substrate (figure 3.6b) ensured that sensor remains firmly attached throughout the experiment while also not adding any extra stiffness duet to this duct tape as the results from this test and test without duct tape which was used to construct SN or wohlner curve and master curve (figure 4.2) has been verified.

The utilized piezoelectric sensor adhered closely to the defined dimensions shown in figure 3.4. This sensor, sourced from PI Physik Instrumente, fell under the category of PIC255/PIC252, and shared properties akin to PZT5. The chosen size is simply the one that is largest provided by PI company and also that the width of the sensor which is 35 mm (figure 3.4) fits perfectly to the width of the asphalt beam which is 50 mm that is used.

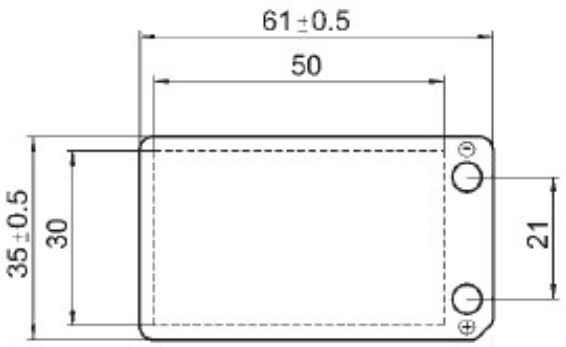
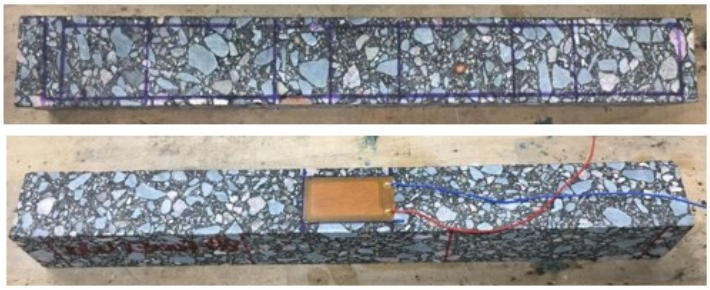


Figure 3.4: Piezoelectric sensor dimensions [30]

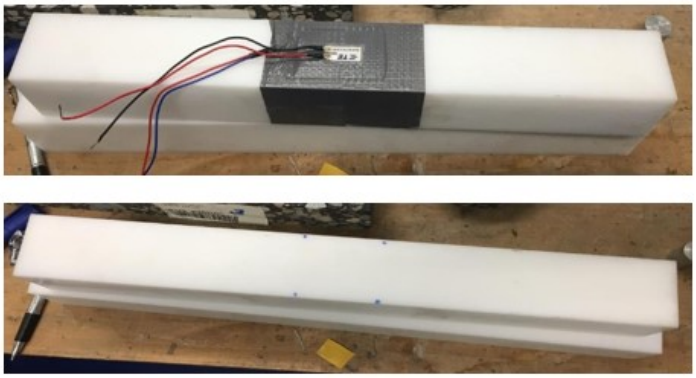
The core of the methodology involved a crucial sensor calibration process. This entailed a thorough examination of the sensor's observable signal behavior under specific frequency and strain conditions, facilitated by a specialized four-point bending machine. The alignment observed between the sensor's output and the expected response underscored the precision of the calibration. Subsequently, a comprehensive dataset was compiled, capturing sensor readings and their corresponding responses, meticulously stored on a USB drive. The collected data underwent in-depth post-processing using Python programming as included in the Appendix, which included the creation of visual signal plots and the extraction of peak-to-peak voltage measurements. These analyses collectively shed light on the sensor's dynamic behavior across various experimental scenarios, providing valuable insights into its performance under varying mechanical conditions.

Following this, a series of experiments was carried out to evaluate the sensor's performance. The sensor was securely placed on the substrate's base and subjected to controlled tests within a four-point bending apparatus. These trials covered a wide range of frequencies, from 1 to 35 Hz, while applying different levels of strain. The experiments were conducted under controlled thermal conditions at 20 degrees Celsius. The data collected from these practical tests provided essential information for thorough analysis, leading to conclusive and enlightening findings.

This methodical arrangement, intricately synchronized with the previously mentioned procedural elements, acted as an effective channel for extensively delving into the piezoelectric sensor's dynamic capabilities and responsive traits within rigorously controlled experimental conditions. The same protocols were applied to both Teflon and Asphalt beams during the four-point bending (4PB) test, and each test except Asphalt was repeated at least five times to ensure the reproducibility and reliability of the collected data. Figures 3.5a and 3.5b depict the comparison between asphalt and Teflon samples with and without the sensor:



(a) Asphalt beam with and without sensor attached



(b) Teflon beam with and without sensor attached

Figure 3.5: Asphalt and Teflon beams with and without sensors

A detailed depiction of the sensor, securely fastened to the asphalt substrate, is presented in Figure 3.6a. The sensor is firmly affixed to the asphalt surface using a resilient double-sided tape, establishing a robust and steadfast connection between these components. The attachment process entails an initial application of a highly adhesive double-sided tape to securely anchor the sensor to the substrate, whether it is Teflon or asphalt. Subsequently, a layer of duct tape is meticulously applied over the assembly, carefully designed to expose the electrical contacts, as shown in Figure 3.6b. This intricate procedural arrangement is clearly illustrated in Figure 3.6.

(a) Closeup of Sensor attached firmly to asphalt

(b) Sensor fixed to asphalt and duct taped leaving electrical contacts open

Figure 3.6: Sensor securely affixed to asphalt with electric contacts exposed.

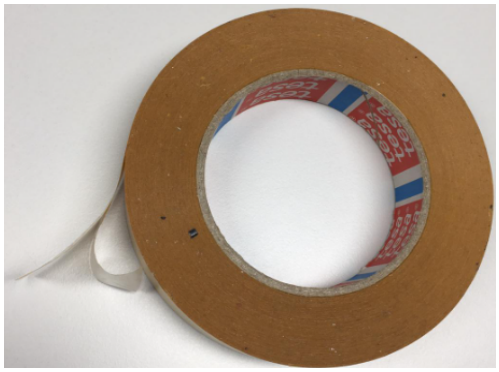
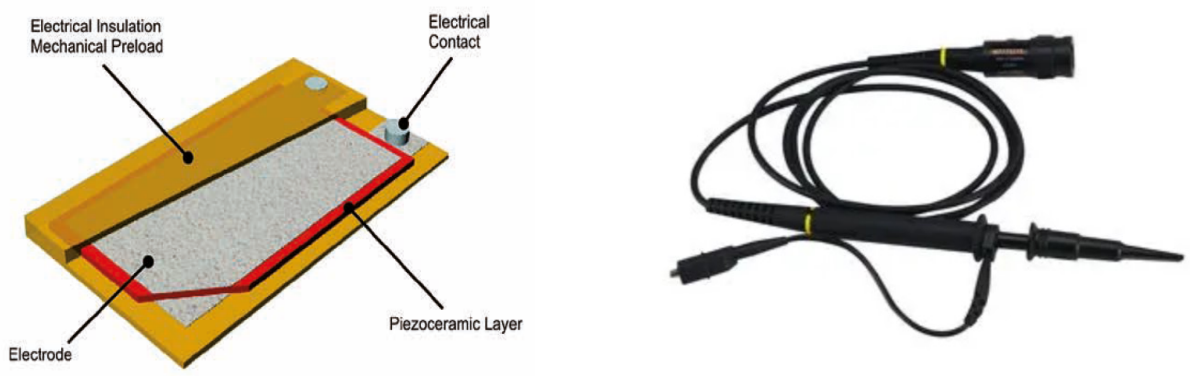


Figure 3.7: Double sided tape

Figure 3.8a showcases the distinct layers comprising the employed piezoelectric sensor. Particularly noteworthy is the role of the ceramic layer in generating voltage as the sensor undergoes bending or conforms to the curvature of the substrate. This fundamental concept has been extensively explained in Section 2.2.0.1 of the study. In the connectivity aspect, the electrical contact links to a wire, which in turn is connected to an oscilloscope through the use of a probe or oscilloscope probe. To maintain signal integrity, the ends of the oscilloscope probe where the wires are attached are thoughtfully insulated, thus minimizing the potential for external interference with the signal output. The evaluation of asphalt's flexural stiffness was conducted both with and without the use of duct tape. The findings indicate that the presence of duct tape and the adhesive layer with sensors does not significantly impact the overall stiffness of the testing process.



(a) PZT sensor and its different layers [30]

(b) Oscilloscope probe

Figure 3.8: Sensor and oscilloscope probe image

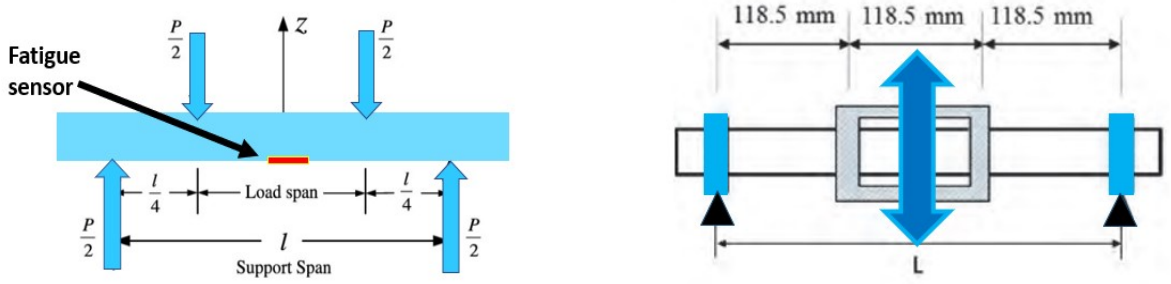
3.1.3. Four point bending test (4PB)

The implementation of the Four-Point Bending (4PB) test setup, featuring the integration of the piezoelectric sensor, is visually depicted in Figure 3.9. This setup incorporates loading clamps designed to facilitate both upward and downward movement, as exemplified in Figure 3.9b. A pivotal aspect of this configuration is the presence of a Linear Variable Differential Transformer (LVDT) positioned at the midpoint. The LVDT functions as a crucial tool for quantifying deformation, which, in turn, is converted into strain using the relevant mathematical relationship, denoted as Equation 3.2.

In preparation for the 4PB tests, the input parameters, which include strain, frequency, and the number of loading cycles, are meticulously configured within the machine's software interface. Subsequently, the loading clamps execute precise movements in accordance with the specified strain levels. This process relies on the well-established direct relationship between deformation and flexural strain, as articulated in Equation 3.2.

The specimens utilized for the 4PB tests adhere to standardized dimensions dictated by European norms, including a length of 400 mm, width of 50 mm, and height of 50 mm. These dimensions align harmoniously with the provisions set forth in NEN-EN 12697-24 [204], which outlines test methodologies for evaluating the fatigue resistance of bituminous mixtures. The fabrication process meticulously adheres to the stipulations outlined by NEN norms. Crucially, these norms require that the effective length between the outer clamps should be no less than six times the maximum value of width or height. For the specimens under examination, this requisite is met, with an effective length of 360 mm, surpassing the stipulated threshold of 300 mm. Furthermore, to fulfill norms, either the width or height must exceed three times the maximum grain size, which in this context, is 16 mm—another condition that is duly satisfied. A further requirement is that the total length should not exceed the effective length by more than 10%, which is indeed met, demonstrated by the calculation:

$$\frac{400 - 360}{400} \times 100 = 10\% \tag{3.1}$$



(a) 4PB test setup **(b)** Schematic of 4PB Test Setup

Figure 3.9: Four-Point Bending Test Setup

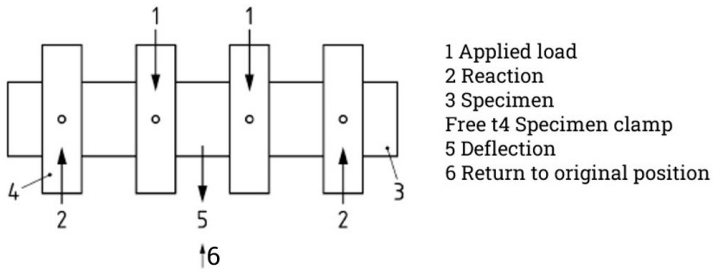


Figure 3.10: 4PB setup according to NEN-EN 12697-24 [204]

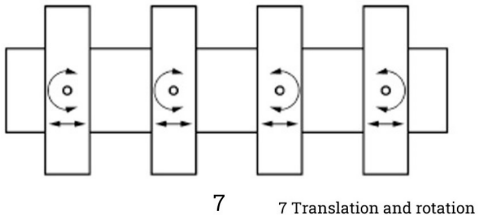


Figure 3.11: 4PB setup according to NEN-EN 12697-24 [204]: Free translation and rotation

The 4PB tests serve as a conduit to achieve three distinct objectives. Foremost among them is the quantification of flexural stiffness, the formulation of predictive master curve as illustrated in figure 4.2, and the construction of Wöhler or S-N curves that illustrate the intricate correlation

between strain and cycles leading to failure in the asphalt concrete beams. Moreover, these tests hold a significant position in confirming the uniformity of voltage outputs. This confirmation encompasses both the piezoelectric and PVDF sensors and applies to both Teflon and asphalt beams. Remarkably, the strategic inclusion of Teflon within the experimental framework aims to amass a substantial dataset—a prerequisite for the comprehensive training of machine learning models tailored for predictive analytics. Notably, this protocol inherently underlines the repeatability of attaining consistent voltage outputs from both Teflon and asphalt beams, further bolstering the data's reliability. The detailed experimental design, presented in Table 3.9, spans a spectrum of sinusoidal strain levels for various frequencies, ranging from 15 µm to 450 µm. This concerted and comprehensive approach effectively encapsulates the multifaceted nature of the 4PB tests, enabling a thorough exploration of the material's dynamic responses and predictive potentialities.

Table 3.9: Four point bending test program

Pulse Width (ms)	Frequency [Hz]	Strain [µmeter/meter]
1000	1	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
500	2	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
400	2.5	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
200	5	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
100	10	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
66.67	15	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
50	20	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
40	25	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
33.33	30	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
28.57	35	15, 50, 100, 150, 200, 250, 300, 350, 400, 450

The equation that links vertical deformation to flexural strain is presented in equation (3.2):

$$\epsilon_t = \frac{12\delta h}{23a^2} \tag{3.2}$$

where δ represents the vertical deformation at the center of the beam, a is the distance between the support and loading clamp, h signifies the height of the beam. SN curve is shown in the figure 4.1a and master curve in figure 4.2 after testing 18 beams.

3.2. Novel structural health monitoring (SHM) methodological approach

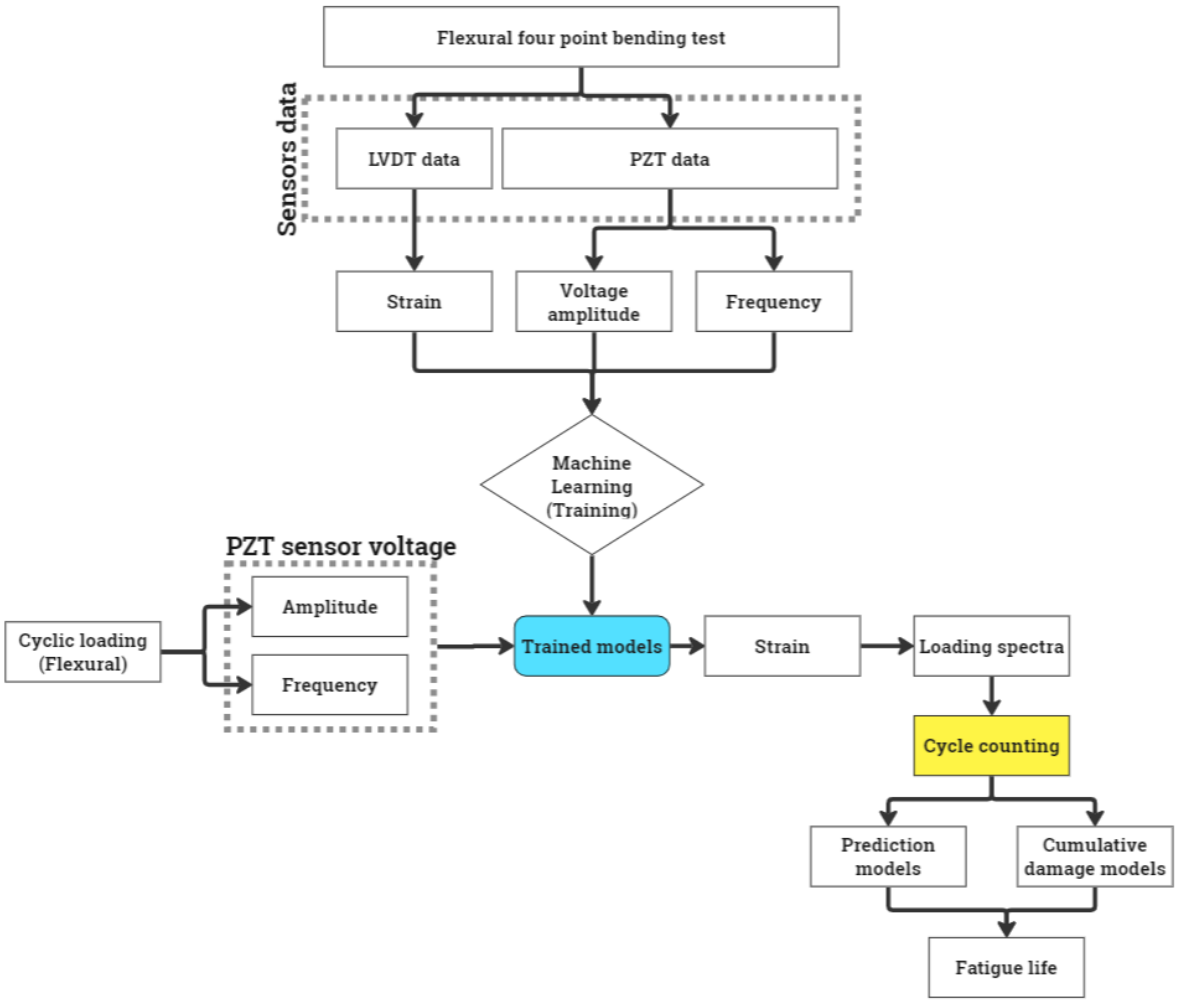


Figure 3.12: General methodology of the research

Depicted within Figure 3.12, the current illustration outlines the main steps used in this research project. It introduces a new way of assessing the health of structures, focusing specifically on the material being studied in terms of predicting its behavior using sensors and simple fatigue life prediction models. The novelty of this research is discussed in detail in section 2.1.

The unique approach involves training machine learning models using data from a test that bends the material at four points. Once these models are trained, they become adaptable to other materials too. This adaptability is possible because the behavior of the material under strain remains consistent, regardless of the specific material. This allows us to use a material like Teflon, which is similar in stiffness to the asphalt beams, for training the models as much data needs to be gathered for ML training purposes. Later, we can validate the models using asphalt beams for a few frequencies.

Importantly, the stiffness of the material being tested doesn't affect the process. This is due to the way the four-point bending machine works, which is designed to maintain a constant level of strain. The machine measures how much the material bends using a device called an LVDT placed at the middle point. This measurement helps us calculate the strain. So, whether a material is stiff or not, the machine adjusts the load to achieve the same amount of bending, ensuring fair comparison.

We choose not to train the models using asphalt beams directly due to asphalt's unique properties. It can behave unpredictably and fail suddenly, potentially damaging the sensors and not providing sufficient data for training the machine learning models.

Once the models are trained, we put asphalt beams through repetitive bending under a consistent wavelike load. The setup for this is shown in Figure 3.9a. We attach a sensor where the bending is most pronounced or where the bending remains steady and horizontal. This sensor is precisely positioned at the midpoint on the underside of the beam.

In conclusion, the outlined methodology showcases both innovative research and thoughtful experimental design. It balances the complexities of material behavior and equipment intricacies, underscoring a comprehensive approach to the research endeavor.

The operational sequence involved in acquiring strain measurements is illustrated in Figure 3.13. Within the context of the four-point bending apparatus, the integration of a Linear Variable Differential Transformer (LVDT) enables the capture of deformations δ, subsequently facilitating the calculation of strain and other crucial parameters, including substrate stiffness, through the 4PB machine software. The mathematical equations required for these calculations have been comprehensively elucidated in Section 3.2.

An alternative way to measure strain involves using a piezoelectric sensor. This sensor produces voltage data, which can be analyzed to find out the signal frequency and the peak-to-peak voltage of the sensor signal. This combination of voltage and frequency creates a special pattern that corresponds to a specific level of strain. By adjusting how much the material is bent, which is related to the given strain level, along with the frequencies used while recording the sensor's voltage responses, a useful set of data can be gathered. This collection of data then forms the foundation for training machine learning models. With this trained group of models, it is ready to estimate strain levels accurately based on any combination of voltage and frequency settings.

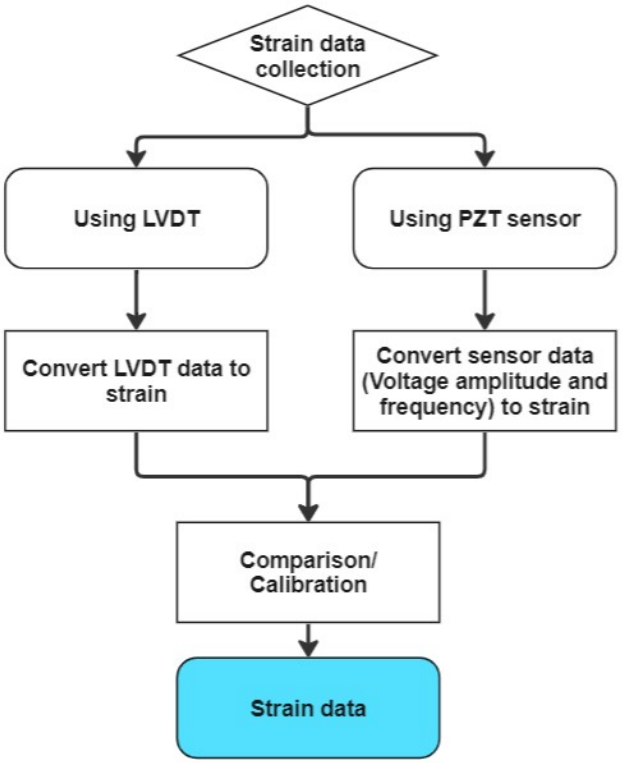


Figure 3.13: Flowchart for getting Strain data

The figure 3.14 presented below captures the prototypical signal derived from a piezoelectric sensor. This illustrative diagram unveils the voltage signal emanating from the sensor, wherein the

temporal evolution showcases alternating positive and negative voltages as time evolves. It's important to note that this figure serves as an illustrative example, while signals stemming from diverse frequencies find elucidation in the appendix refer to the appendix for further details].

This signal encapsulates several pivotal insights, including the amplitude between the signal's peak and trough, the sensor's response to mechanical stimuli, both the sensor itself and the substrate and the signal's frequency, aligning with the input frequency administered to the machine.

In this section, various methods are explained, each presenting a different way to predict strain. Regardless of the mix of voltage and frequency used, the main aim here is to create strong models that work well not only in labs but also in real-world situations.

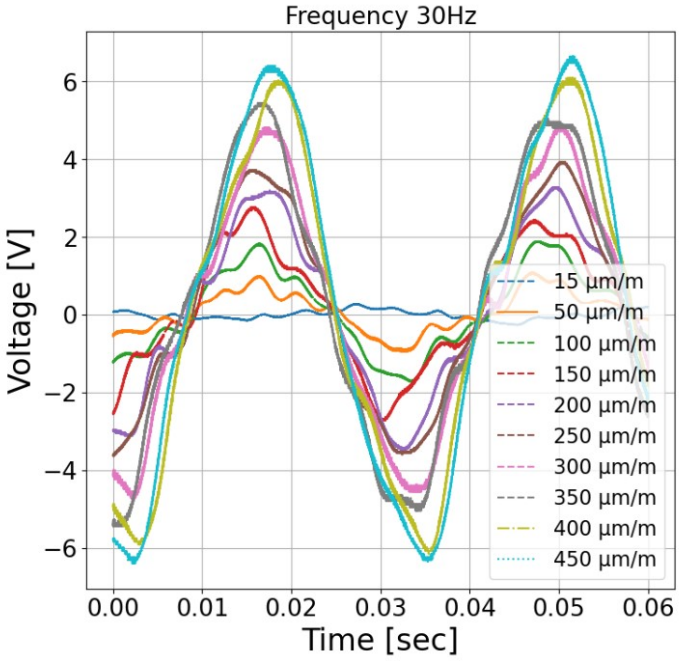


Figure 3.14: Piezoelectric sensor (PZT) voltage vs time curve

3.3. COMSOL numerical analysis

COMSOL multiphysics 6.0 has been used for numerical analysis in order to model Teflon beam under four-point bending to verify whether this will produce similar pattern as also observed during 4Pb test. The four-point bending has been already explained in section 3.1.3. This finite element program allows complex numerical analysis of different types of problems allowing multi-physics simulation and calculation. This is the reason this software has been used for this research as it allows mechanical and electrical multi-physics numerical solutions since in this research Teflon bar is mechanically pushed to register voltage from a piezoelectric sensor attached in the middle of the Teflon bar.

The model that has been built in COMSOL interface is illustrated in figure 3.18.

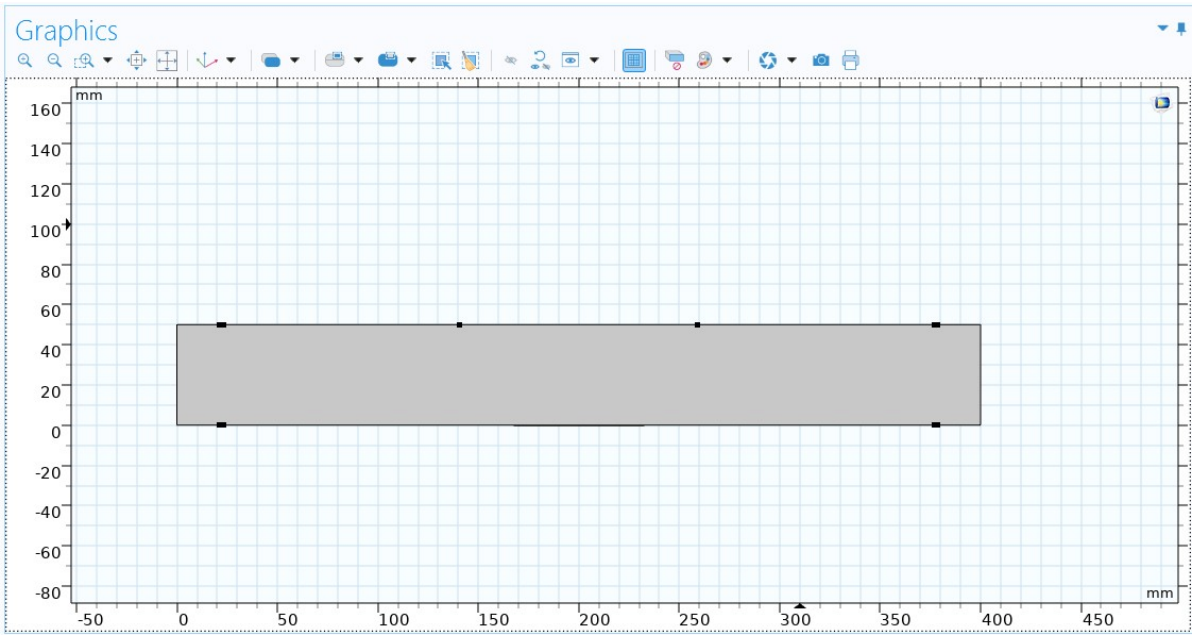


Figure 3.15: Model built in COMSOL

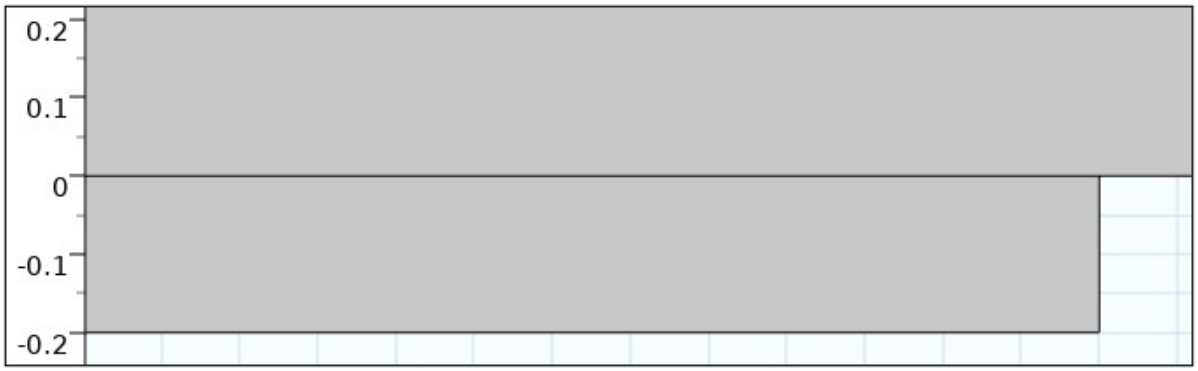
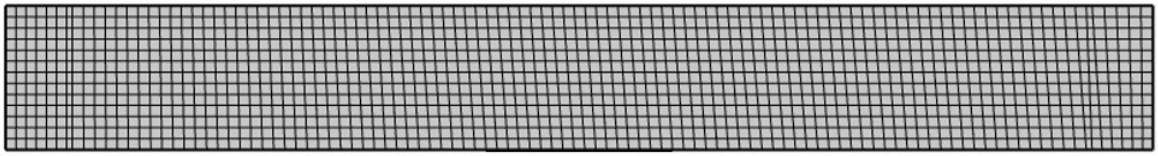


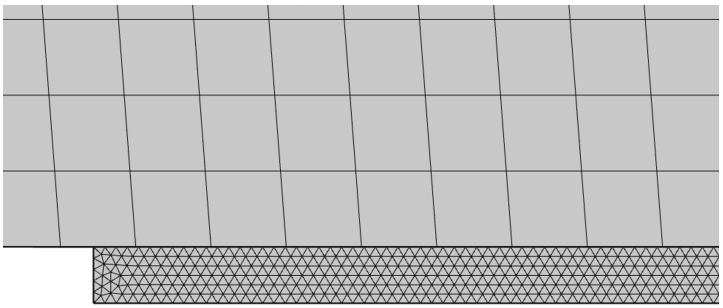
Figure 3.16: Model zoomed in

The model has been illustrated in the figure 3.18. The sensor as illustrated in figure 3.4 is attached to the bottom side of the substrate in the middle. Figure 3.16 gives more detail on how the sensor is attached to the substrate with a thickness of 0.2 mm and a length of about 61 mm. Thickness is basically the layer thickness where electrical charge is generated and not the complete sensor thickness. Above the sensor and substrate, on each side loading clamps are shown which move up and down as in the four-point bending test setup. On the far side, we see on either side of the substrate supporting clamps which are behaving as figure 3.11.

Two different types of mesh is used as shown in figure 3.17b. As figure 3.17a, mesh used for sensor can not be seen clearly so a close up view can be seen in figure 3.17b instead.



(a) Mesh overview



(b) Close-Up View of Mesh Structure

Figure 3.17: Mesh used for COMSOL model

In the context of substrate properties, the selection of Teflon material was employed due to its stiffness closely resembling that of asphalt. The underlying theory posits that the material composition of the substrate should not significantly impact the sensor voltage output, provided that the correct force is applied to the substrate in regions containing rollers or loading clamps. It is noteworthy that a stiffer substrate necessitates a higher applied force to induce equivalent deflection at the center of the substrate. This aligns with practical experimentation, as elucidated in Section 3.1.3, wherein an LVDT (Linear Variable Differential Transformer) is positioned at the midpoint of the beam to measure deflection. This deflection data is subsequently translated into midspan flexural strain, as the test procedure entails controlled strain testing. Consequently, in order to achieve the desired midspan deflection, the testing machine will exert a greater force on materials characterized by higher stiffness as compared to those with lower stiffness. Consequently, the choice of substrate material is rendered inconsequential. This does not include material behaviour or cracking pattern as they are not of concern in this research.

This inherent property holds significant advantages for the present research endeavor, as Teflon can be effectively employed as a substrate material due to its similarity in stiffness to asphalt concrete. This allows for the generation of extensive datasets, incorporating a minimum of fourfold repetitions, which can be instrumental in training machine learning models. Furthermore, these well-trained models can subsequently be employed for the analysis of sensor data obtained from asphalt beams.

Hence, within the scope of this numerical analysis, the deliberate selection of Teflon as the substrate material aligns with the research's central focus on sensor data, recognizing its paramount significance in the investigative process. By ensuring the compatibility and verification of data obtained through COMSOL simulations with corresponding experimental results, the acquired dataset from four-point bending experiments utilizing Teflon as the substrate can be confidently harnessed for the training of machine learning models. This strategic approach underlines the research's commitment to robust data-driven analysis and model development, ultimately enhancing the research's overall efficacy and reliability.

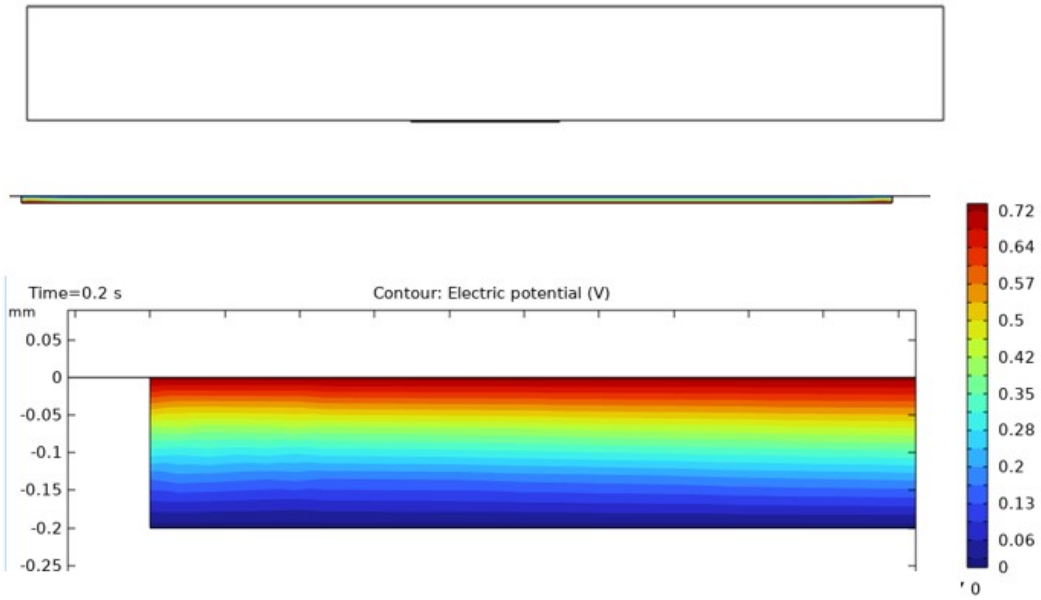


Figure 3.18: Typical look of model with sensor

In figure 3.18 above, the model is shown, and then below that only the sensor is shown and below that close-up view of the sensor is shown. The red area shows 0.72 electric potential (V). This color shifts so either the top side or bottom side becomes red and the other side becomes 0 volts depending on how the sensor is bent by the substrate.

As explained earlier, the substrate material is Teflon with similar stiffness to the asphalt beam used and the sensor is PZT-4 with few of its properties changed according to the catalog of PI piezoelectric sensor PIC255 [30, 205].

Compliance matrix, Voigt notation

Symmetric

1.604e-011[1/Pa]	-5.678e-012[1/P...	-7.444e-012[1/P...	0[1/Pa]	0[1/Pa]	0[1/Pa]
-5.678e-012[1/P...	1.604e-011[1/Pa]	-7.444e-012[1/P...	0[1/Pa]	0[1/Pa]	0[1/Pa]
-7.444e-012[1/P...	-7.444e-012[1/P...	1.903e-011[1/Pa]	0[1/Pa]	0[1/Pa]	0[1/Pa]
0[1/Pa]	0[1/Pa]	0[1/Pa]	4.595e-011[1/Pa]	0[1/Pa]	0[1/Pa]
0[1/Pa]	0[1/Pa]	0[1/Pa]	0[1/Pa]	4.595e-011[1/Pa]	0[1/Pa]
0[1/Pa]	0[1/Pa]	0[1/Pa]	0[1/Pa]	0[1/Pa]	4.595e-011[1/Pa]

OK Cancel

Figure 3.19: PZT-4 Compliance matrix

Coupling matrix, Voigt notation

0[C/N]	0[C/N]	0[C/N]	0[C/N]	5.840e-010[C/N]	0[C/N]
0[C/N]	0[C/N]	0[C/N]	5.840e-010[C/N]	0[C/N]	0[C/N]
-1.750e-010[C/N]	-1.750e-010[C/N]	3.95e-010[C/N]	0[C/N]	0[C/N]	0[C/N]

OK Cancel

Figure 3.20: PZT-4 Coupling matrix

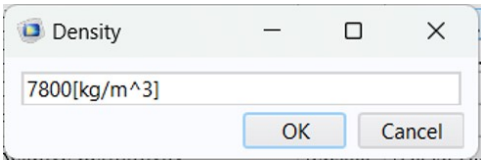


Figure 3.21: PZT-4 density

Results from the numerical analysis is present in section 4.2 of this thesis.

3.4. Machine learning (ML) pipeline

Machine learning and artificial intelligence a powerful and innovative tool that can be used for a range of applications which includes classification problems such as classifying what objects are or regression problems such as predicting asphalt performance based on material design or as used in this research which is predicting strain based on voltage and frequency combination. When any task is performed based on data instead of logic that is simply the definition of machine learning [206]. It is therefore important to have good data and a good machine learning model meaning that the model is not too rigid and is more flexible or in other words we tolerate some inaccuracy but we have a good fit between predicted and actual data so we don't risk of under-fitting or over-fitting [206] as also explained in section 3.4.4. Figures 3.25 and 3.26 show what these different types of models look like. Therefore, training of the models has been done in such a way as to ensure a good fit. In this study, an analysis of the data derived from the piezoelectric voltage measurements revealed a high degree of correlation when examining the voltages associated with various frequencies. The calculation of this correlation was executed using Pearson's correlation factor, as outlined in Equation (3.3), which was instrumental in establishing a correlation matrix. A corresponding heatmap was generated and is presented in Appendix A. It is noteworthy that prior research, conducted by various scholars, has already demonstrated that high correlation, leading to the issue of multicollinearity (also known as extrinsic aliasing), does not exert any adverse impact on the predictive quality or Machine Learning (ML) regression models, as documented in relevant studies [207, 208].

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{(n\sum x^2 - (\sum x)^2)(n\sum y^2 - (\sum y)^2)}} \tag{3.3}$$

In coming sections, it is explained how machine learning models are trained and used to predict strain values. This section deals with the machine learning pipeline and the algorithms or the ML-models that have been used are discussed in detail in section 2.7. 13 machine learning algorithms have been used in this research and their results compared in terms of accuracy by plotting bar graphs for coefficient of determination (R^2), root mean squared error (RMSE) and mean absolute error (MAE). Also in the end comparison is made on how effective these models are if voltage recorded from a piezoelectric sensor is recorded after bending asphalt beams under four-point bending. These machine learning algorithms and the branch of machine learning where they are taken from are summarized in the figure 3.23:

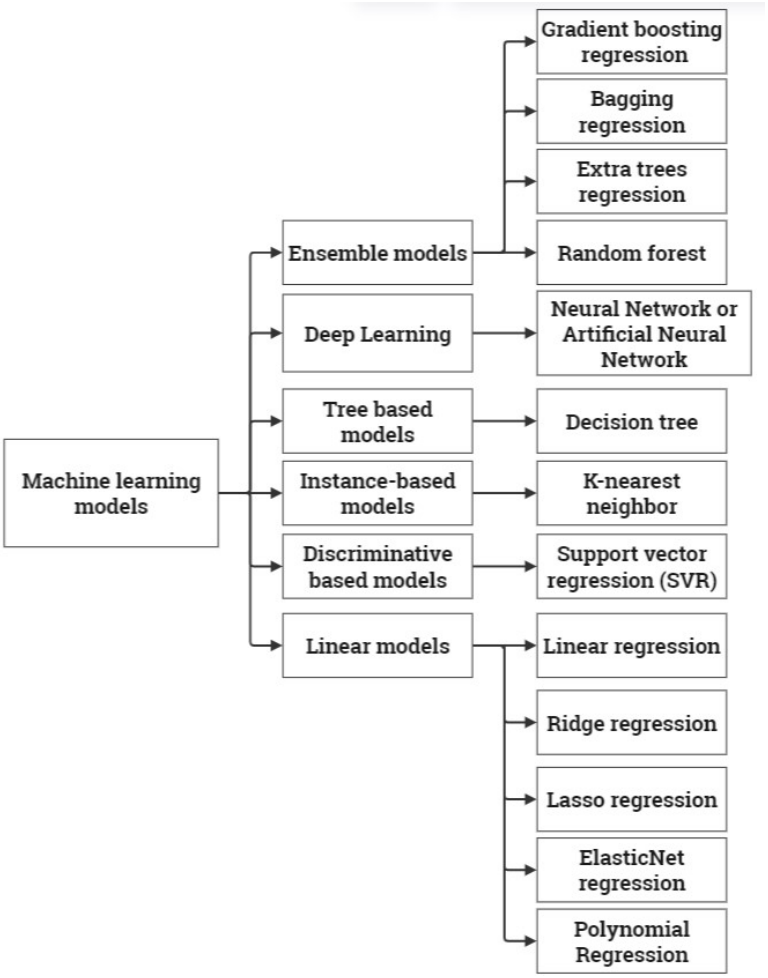


Figure 3.22: Machine learning algorithms used

Each of the models or algorithms above is explained in great detail in section 2.7 where links are provided for more reading. In this section, however, the machine learning pipeline is explained which is also summarized in the flowchart 3.23 that has been applied for this research.

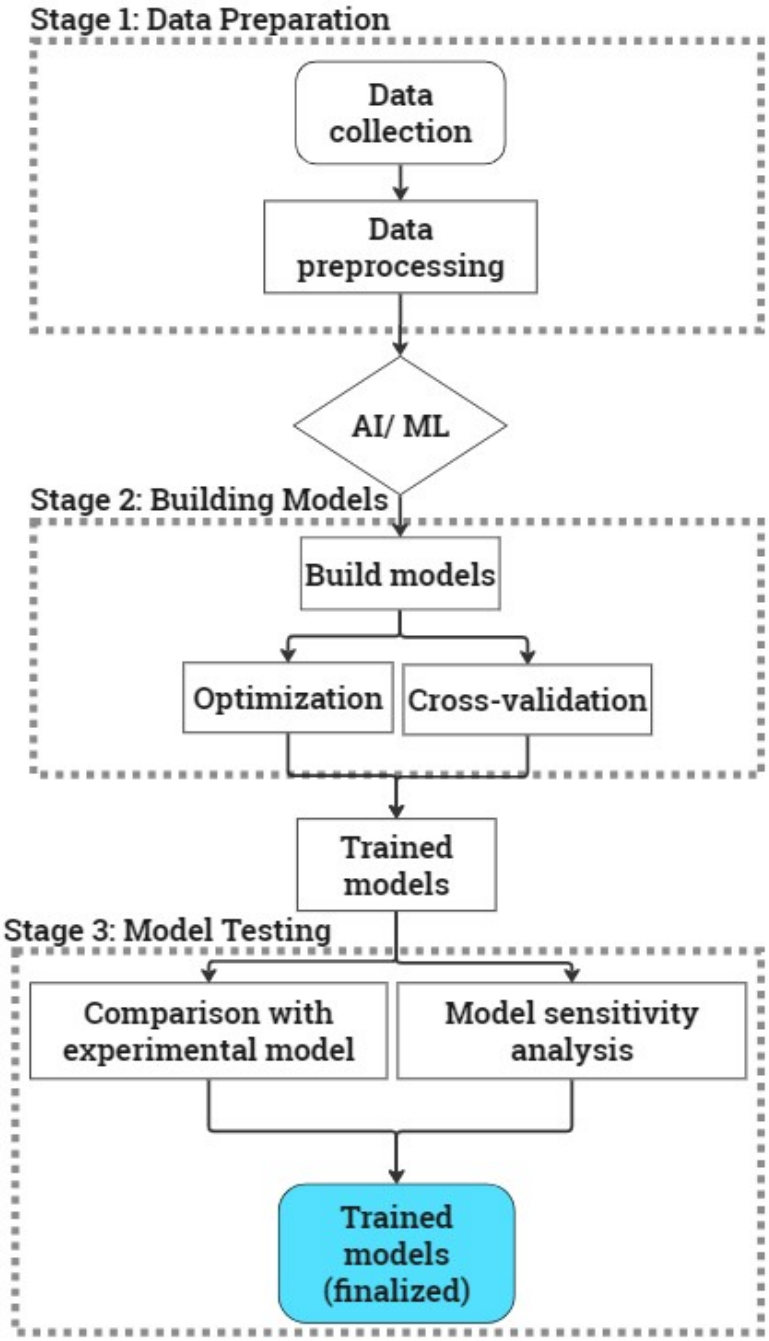


Figure 3.23: Machine learning pipeline

3.4.1. Data preprocessing

Preprocessing involves cleaning and preparing data by addressing missing values, outliers, and imbalanced data, while feature engineering creates new features from existing data through techniques like combining, transforming, or applying domain knowledge [209]. Data preprocessing is an important step towards machine learning as decisions at this stage have an effect on the model's predictive behavior [210]. Hence, feature engineering is important to be done to get the best quality of data for any machine learning model. It includes the following steps:

- Data cleaning
- Data reduction

- Data integration
- Data transformation
- Data creation
- Data selection

Depending on the model, some or all of the above steps can be applied to get the best quality model.

The first step is data cleaning which as the name suggests is the step where data are cleaned in other words the two steps are data qualitative error detection and error repairing [211] such as removing inconsistencies, filling in missing values, and smoothing noises [212]. Data reduction techniques also lie under data cleaning since they trim and filter data from a large dataset while preserving key information and key essence of the data. Data integration mitigates variable conflicts.

Data transformation constitutes a pivotal step encompassing both data preprocessing and feature engineering, entailing alterations aimed at enhancing data quality or engendering novel analytical features [213]. In the realm of data transformation, datasets undergo processes of scaling and encoding [212]. Scaling procedures are invoked to rectify substantial deviations [212]. Moreover, datasets frequently comprise a medley of numerical and categorical variables, necessitating the translation of nominal data into numerical representations. In the feature creation phase, variables more congruent with the model's objectives are synthesized [214]. During the feature selection phase, the ultimate dependent and independent variables are judiciously curated. It's imperative to acknowledge that these processes exhibit an iterative nature and eschew strict adherence to a predetermined sequence [213]. Across the modeling continuum, data preprocessing and engineering are recurrently and judiciously applied. The ensuing subsection offers a comprehensive elucidation of these aforesaid methodologies.

3.4.1.1. Data reduction and cleaning

Persistent observations have consistently shown that machine learning (ML) problems are highly prone to data contamination, highlighting this vulnerability even when robust methodologies are employed [215, 216, 145, 217]. The data gathered from the four-point bending tests comprises voltage measurements, flexural strain data, and frequency values. It should be noted that there are only two features such as voltage and frequency that are used as features to train ML models to get strain as output.

No feature has been removed as no parameters have been used for bookkeeping or as not many samples have been used. Also since all features were collected from one four-point bending machine there is no or little effect of the used substrate on the result data. Regression coefficients have been used to determine dependent variables. Since the data source is from a unique source, there is no conflict since different sources may cause conflicting conclusions [212]

filling in 30Hz , 200 μm/m.

3.4.1.2. Data integration

Data integration constitutes the systematic consolidation of data originating from heterogeneous sources, aiming to furnish end-users with a comprehensive and unified data representation [218]. This pivotal procedure, intrinsic to the domain of data preprocessing, is orchestrated by data architects to automatize the seamless transmission of data from source entities to designated target systems, thereby guaranteeing a holistic and all-encompassing data outlook.

Since data used to train machine learning models are from one source and from one 4PB machine, no data integration is required.

3.4.1.3. Splitting

Prior to advancing to the subsequent stage, it is imperative to bifurcate the dataset into distinct training and testing subsets as shown in the figure 3.26:

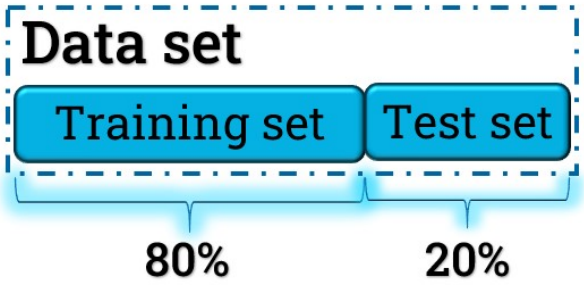


Figure 3.24: Splitting of data set

The process of fitting a regression model mandates the segregation of the dataset into two discrete sets, as elucidated by Stone et al. in their seminal work on cross-validatory choice and assessment of statistical predictions (1974) [219]. The fundamental principle underlying this procedure is to formulate a hypothesis leveraging the training set and subsequently subject it to rigorous scrutiny using the testing set [219]. The partitioning process involved allocating 80% of the dataset for training purposes and reserving the remaining 20% for testing, a customary approach in statistical analysis. In this critical phase, a meticulous examination of the dataset's distribution was conducted to ensure it retained its original characteristics. It's important to highlight that the exact same dataset was used for both statistical and machine learning analyses, ensuring the consistency and reliability of the analyses.

3.4.2. Data transformation

Data transformation is unnecessary in this case as there are no categorical data, and the chosen model does not face challenges related to varying data ranges.

3.4.3. Feature engineering

Feature scaling is unnecessary because all features share the same source and characteristics. Additionally, feature creation is not required as the existing features provide sufficient information for accurate predictions.

3.4.4. Regression analysis

Regression is a prevalent statistical method employed for modeling, wherein it establishes a mathematical connection between a response variable (y) and explanatory variables (x) to describe systems [220]. Regression analysis, such as linear regression, represents one of the foremost and most comprehensive statistical and machine learning algorithms [221].

Regression analysis offers a multitude of valuable advantages and functionalities in the realm of statistical analysis and predictive modeling [222]. It serves as a powerful tool for not only making predictions but also for conducting estimations and providing explanations. One of its fundamental roles lies in the ability to elucidate intricate relationships and dependencies within datasets, particularly in terms of scrutinizing the correlation between variables [223]. Through regression analysis, it becomes feasible to delve into the intricacies of how independent variables exert influence on dependent variables. This encompasses not only quantifying the strength and direction of such influence but also discerning the existence and nature of relationships between these variables. As a result, regression analysis serves as a robust framework for uncovering the underlying dynamics and associations present in data, facilitating a comprehensive understanding of complex phenomena in various fields of study, from economics to the natural sciences and engineering. Regression analysis involves the process of fitting a line among a set of observations in a manner that minimizes the total sum of residuals [224].

The process of conducting a regression analysis involves a sequence of fundamental steps. These steps encompass the meticulous definition of the dataset under investigation, the formulation of a mathematical model that can accurately describe the data, a comprehensive assessment of the model's appropriateness and how well it aligns with the observed data, and the strategic

application of a fitting technique to optimize the model's parameters for a robust and reliable analysis. Data used in this research comes piezoelectric sensor from a point bending test. The general objective is to get a model that is neither overfitting nor underfitting [220].

In machine learning, under-fitting occurs when the algorithm fails to capture the inherent patterns in the data, resulting in suboptimal model performance [225]. To address this issue, one can employ a mathematical model that adequately matches the data's complexity and ensures the inclusion of a sufficient number of relevant features in the analysis.

On the other hand, over-fitting happens when the algorithm excessively conforms to the training data, potentially leading to poor performance when dealing with new, unseen data, particularly in complex classification or multi-dimensional scenarios. To mitigate over-fitting, established practices involve segregating the dataset into separate training and testing sets and utilizing cross-validation techniques to evaluate the model's ability to generalize [226].

Figure 3.25 and 3.26 provide a visual representation of various fitting scenarios, encompassing both model training and prediction. Multicollinearity, a phenomenon where two or more predictor variables, often referred to as independent variables, exhibit high correlation, can lead to extrinsic aliasing or multicollinearity [227]. This phenomenon may result in overfitting, rendering models less stable and less interpretable [227]. However, in the context of utilizing a regression model for prediction, the presence of multicollinearity has no impact [227, 228]. Its effects are primarily confined to individual coefficients and associated p-values [227, 228].

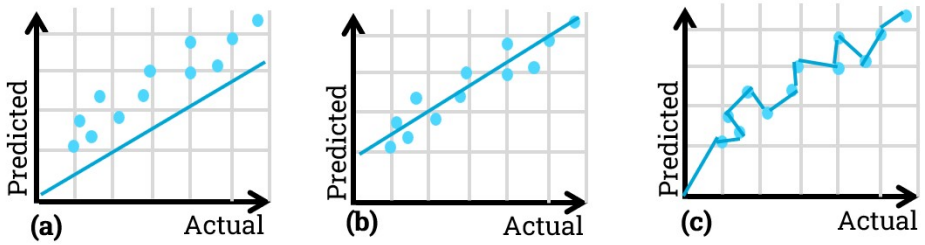


Figure 3.25: Different fitting examples (a) Under-fit, (b) Good-fit, (c) Over-fit

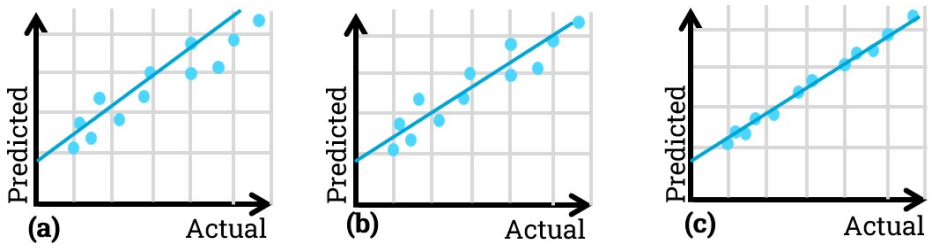


Figure 3.26: Different prediction fitting examples (a) Under-fit, (b) Good-fit, (c) Over-fit

The coefficient of determination, commonly denoted as R-squared (R^2), serves as a valuable statistical metric for assessing the equilibrium between the phenomena of over-fitting and underfitting within a given model [229].R-squared, an essential statistical metric, quantifies the degree to which changes or fluctuations in the dependent variable can be explained or attributed to changes in the independent variables. Notably, R-squared augments as additional predictor variables are incorporated into the Multiple Linear Regression (MLR) model [230, 229]. The values of R-squared are bounded within the interval [0, 1]. In this research R-squared is within range of 68% to 98%.

$$R^2 = 1 - \frac{X}{Y} * 100 \tag{3.4}$$

$$X = \sum_{i=1}^{N} (y_n - f_n)^2 \tag{3.5}$$

$$Y = \sum_{i=1}^{N} (y_n - \bar{y})^2 \tag{3.6}$$

The regression analysis encompassed the selection of two primary models: Multiple Linear Regression (MLR) and Multiple Polynomial Regression. These models were employed for statistical analysis and the generation of voltage-time curves based on data acquired from the sensor [231, 232, 233].

In the realm of machine learning analysis, a total of 13 distinct models were chosen. The application of machine learning techniques in pavement engineering is not a novel concept and has previously been utilized for prediction, with MLR being particularly prominent due to its simplicity and ease of interpretation [231, 232, 233, 234].

Support Vector Regression (SVR), Random Forest (RF), and Gradient Boosting Regression (GB) are all well-established algorithms within the field of pavement engineering. However, the innovative aspect lies in the utilization of machine learning to derive strain values from a combination of voltage and frequency data. This novel approach enables sensors to function as fatigue nodes, wherein the collection and processing of voltage and frequency data facilitate the computation of strain values. These strain values, in turn, can be employed by phenomenological models to predict fatigue life. Presently, this approach is confined to laboratory-based four-point bending (4PB) experimental tests, but its extension to real-world field studies is readily achievable [235, 236, 233].

Subsequent to model construction, additional steps were undertaken, including optimization, cross-validation, and the comparative analysis of machine learning model outputs against experimental results while also looking into model sensitivity, even though, in the current context, the simplicity of the model is evident, given that only two features were utilized to predict the target variable [231, 232, 233, 234].

3.4.5. Optimization

Model optimization, often referred to as tuning, constitutes a pivotal phase in the realm of machine learning [237]. Hyperparameters, which are parameters employed to fine-tune a machine learning model or minimize an objective function, play a central role in this process [238]. These hyperparameters were predetermined, and the algorithm autonomously identified the optimal configuration. The optimization of hyperparameters is achieved by employing the following equation [238]:

$$x^* = \arg\min_{x \epsilon \chi} (f(x)) \tag{3.7}$$

where:
f(x): The objective function to be minimized
x^* Set of hyperparameters which yields to minimize the function

However, for this research negative RMSE (Root Mean Squarred Error) is minimized. Another way is to minimize the negative of the R-squared so the model with the highest R-squared will be the least negative R-squared and the best hyperparameter for that particular model. Same applies to all the models and tuning hyperparameters.

Bayesian Optimization which uses the probabilistic method is superior to random and grid search [239]. However, in this research Gridsearch algorithm is used as it is the traditional way, and only two features such as voltage and frequency are been used. there is no further need to use more complex optimization techniques.

Furthermore, alongside the optimization of hyperparameters, loss functions hold a crucial significance in enhancing the performance of machine learning models [240].

In machine learning, hyperparameters are crucial settings that control the learning process. Hyper-parameters is essential in machine learning which impacts learning efficiency and model quality results. Selecting optimal hyperparameters is therefore a critical step in training a model, as they guide the learning algorithm in finding the best parameters to map input features to target labels, enhancing model performance.

The following subsection elucidates several loss functions employed in the analysis, which were consistently applied to all models for sensitivity analysis. However, for hyperparameters tuning, RMSE is minimized.

3.4.5.1. Loss function

Loss functions used in this research are coefficient of determination (R^2), root mean squared error, mean absolute error, quantile loss, and expected loss. Loss functions help machine learning models to become more accurate and efficient [240].

- coefficient of determination (R^2) [241]
- Mean absolute error (MAE) [242]
- Mean squared error (MSE) [243]

The Root Mean Squared Error (RMSE) which resembles the Euclidean distance [244] as it can also be seen in the equation 3.8 is similar to the distance from the prediction f_n to the actual value y_n and finally it has been re-scaled by dividing with factor N allowing estimation of the standard deviation.

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N}(y_n - (f_n)^2)}{N}} \tag{3.8}$$

where:
n is just a variable
y_n is actual value
f_n is the prediction value
N is the number of data points

All models in the analysis were trained and optimized using a coefficient of determination (R^2). Afterwards, these models were trained using all the loss functions.

Quantile loss is defined as the equation 3.9:

$$Quantile = \sum_{n=1}^{N} w_{n,\alpha}|y_n - f_{n,\alpha}| \tag{3.9}$$

Where

$$\begin{cases} 1-\alpha & \text{for } y_n < f_{n,\alpha} \\ \alpha & \text{for } y_n > f_{n,\alpha} \end{cases}$$

It should be noted that in the equation above asymmetric weights $w_{n,\alpha}$ are balanced by α. The α parameter spans the interval from 0 to 1 and serves to impose penalties on predictions [245]. For values of α above 0.5, the over predictions are penalized. Below 0.5, the under predictions are penalized, and 0.5 penalizes equally.

Expectile loss is given in the equation 3.10:

$$Expectile = \sum_{n=1}^{N} \omega_{n,\tau}(y_n - f_{n,\tau})^2 \tag{3.10}$$

Expectile loss is similar to quantile loss with only difference that in expectile loss distance $(y_n - f_{n,\tau})^2$ is a quadratic term which leads to asymmetric least squares [245]

[246]

3.4.6. Cross validation

One of the most common approaches or methods for generalizing the models is to do cross-validation [206]. There are different types of cross-validation as also shown in the figure 3.27. When the data set is divided into smaller subsets, this will cause the distribution to change. For cross-validation models are trained in smaller subsets and the results are then averaged [206].

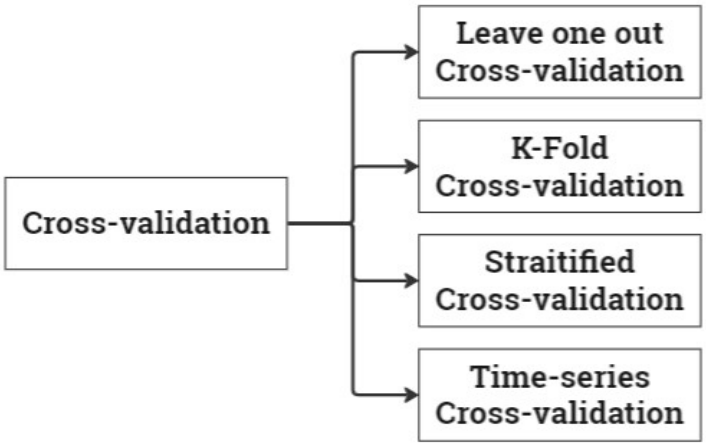


Figure 3.27: Cross-validation types

K-fold cross-validation has been used in this research. K-fold basically split the data into k-equal subsets [206, 247]. In this research, 5-fold cross-validation has been used for all the datasets. The prediction function was learned using k-1 folds, the left-out fold is used for the test. This process is repeated as the validation set will be exchanged for training sets until all the k-samples have been used for training and testing [206] as also shown in figure 3.29.

. This is the last step of the optimization as we now have hyperparameters tuned and cross-validated. Now next will be to see the model sensitivity by doing a sensitivity analysis.

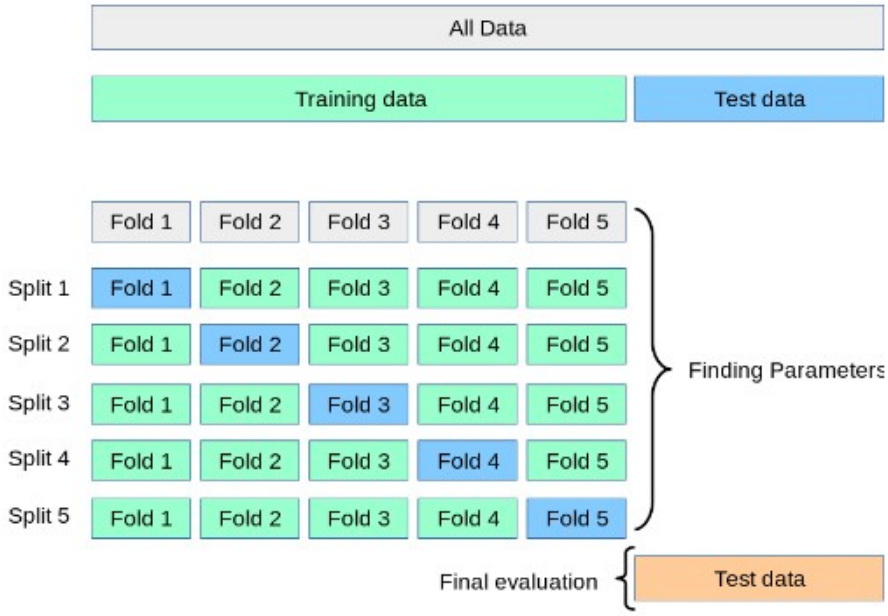


Figure 3.28: K-fold cross-validation [248]

3.4.7. Model sensitivity

Machine learning models are dependent on the data given as it is simply a black box with data coming in and results going out. It is therefore essential to interpret the results carefully in order to understand what and how results are affected and understand model sensitivity. Therefore, explainable machine learning techniques are required such as SHAP method which was developed by Lloyd Shapley [249] and it is also known as **SH**apley **A**dditive ex**Pl**anations. This method is based on cooperative game theory which helps to increase transparency and interpretability of ML models. With this technique, the contribution of a specific member of a coalition which produces value can be understood and also the contribution of a member of a coalition is calculated by comparing differences between the values of this coalition with and without the member with the difference showing the marginal contribution of the member [234]. Shapely value is simply the mean contribution of the member considering all the possible combinations in which it is included. This method however does not judge the quality of prediction but rather shows the contribution or importance of each feature which it considers as ”players” and their end contribution as ”prize”.

There are several ways for visualizing SHAP values [250] such as summary plots, force plots, dependence plots, bee swarm plots, etc. Since only two features are used, it is chosen for summary plots as shown in the next chapter ”Results”. One of the SHAP summary plots is shown in figure 3.29. On the left-hand side features are listed and ranked by their importance [245]. The mean absolute value of SHAP values for each observation is then what defines the order and the color bar represents each feature so blue stands for the minimum value of the feature in the dataset and red for the maximum and everything in between is shown by the gradient change from blue to red. There are categorical features used in ML models hence there are no grey colors. X axis represents the impact of the data point on the model so the more it shifts right the higher and positive impact or contribution it has on the mode and the more negative it goes the more negative impact or contribution it has. In the middle represents that individual contribution is zero.

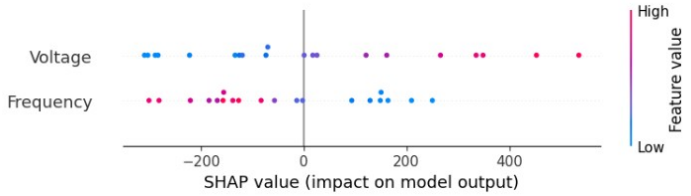


Figure 3.29: SHAP summary plot Neural Network (NN)

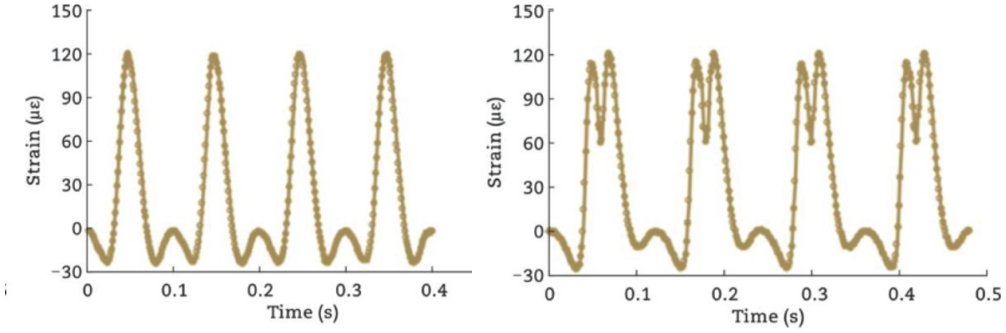
In the following sections, Fatigue life prediction models to give an estimate of fatigue life and models that allow construction damage curves are given. It is important to realize that all of these can be automated in the future. Current results for constant strain test shows good results however they can be further improved and a new model based on using a sensor can be proposed.

3.5. Fatigue life prediction

Following the training of machine learning models, they can serve as predictive tools for strain estimation based on inputs of voltages and frequencies. It should be noted that these machine learning models are part of monitoring models and the methods or models that are discussed in coming sections can also be integrated with machine learning models to form one framework. However, this is not done in this research as that will also provide similar results to this and also since the scope is academic instead of commercial. It's important to note that this process currently operates within the framework of supervised learning, as previously mentioned, with the ultimate aspiration of achieving full automation in the future. The underlying concept entails the utilization of piezoelectric sensors to capture voltage and frequency data, which is subsequently translated into strain values employing machine learning models. This strain information, in conjunction with the count of loading cycles, holds the potential to forecast fatigue life. In an ideal scenario, the count of loading cycles could be extracted and recorded from the voltage-time curve, a product of sensor data, using the rainflow cycle counting technique. However, it's worth mentioning that, in the current context of this research, the count of cycles until failure is pre-known. Nevertheless, in prospective applications where the sensor operates as a sensor node, the rainflow cycle counting method can be employed to accurately register the count of cycles in addition to recording voltage and frequency data.

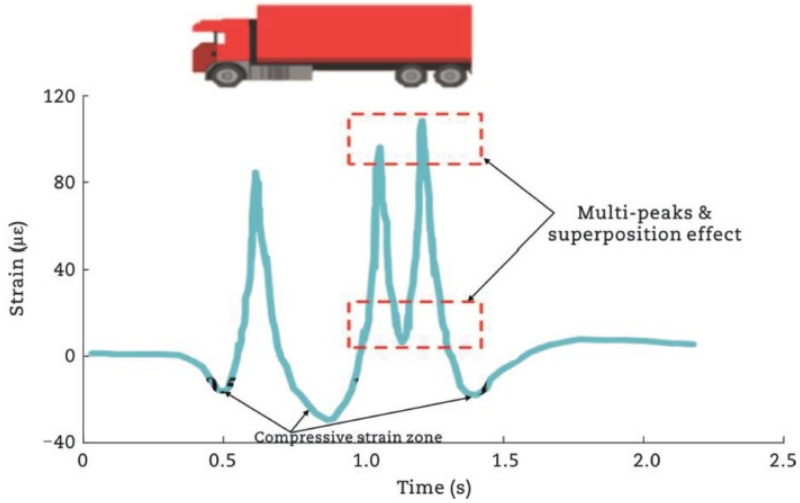
Utilizing strain values, the count of cycles until failure, and various material properties, it becomes feasible to predict the fatigue life of asphalt within the tolerated margin. This predictive capability can be harnessed to construct a damage curve, employing phenomenological models and the SN-curve (commonly known as the Wöhler curve).

Before explaining the models that have been used in this research to show the potential of this sensing method. It is important to realize that in the real field, the loading wave looks something similar like in figure 3.31a and figure 3.30 if the axle loads are driven accordingly. A piezoelectric sensor is quite sensitive to a range of frequencies and strain values so it can be captured very well by the sensor. This research is limited to laboratory research hence only sinusoidal loading is applied with constant frequency at different strain levels rather than applying complex loading on substrate.

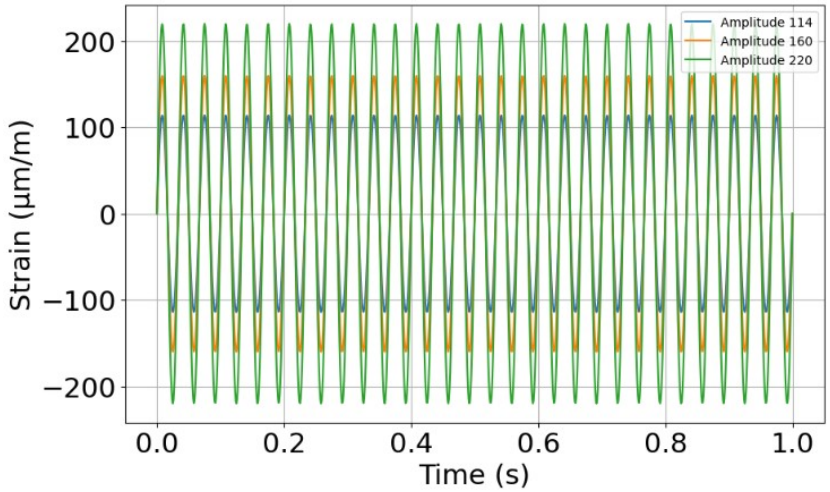


(a) Single-axle wave **(b)** Tandem axle load

Figure 3.30: Loading waveform [109]



(a) Strain responses under single and tandem axle loads, similar to the truck loading scenario [143]



(b) Typical sinusoidal loading applied in 4PB test

Figure 3.31: Traffic vs. laboratory strain waveform

3.5.1. Using SN curve

One of the well known and standardized methods of determining fatigue life is when the stiffness becomes half of the initial value. This value has been used as N_f in future figures and calculations.

A typical fatigue line equation looks like:

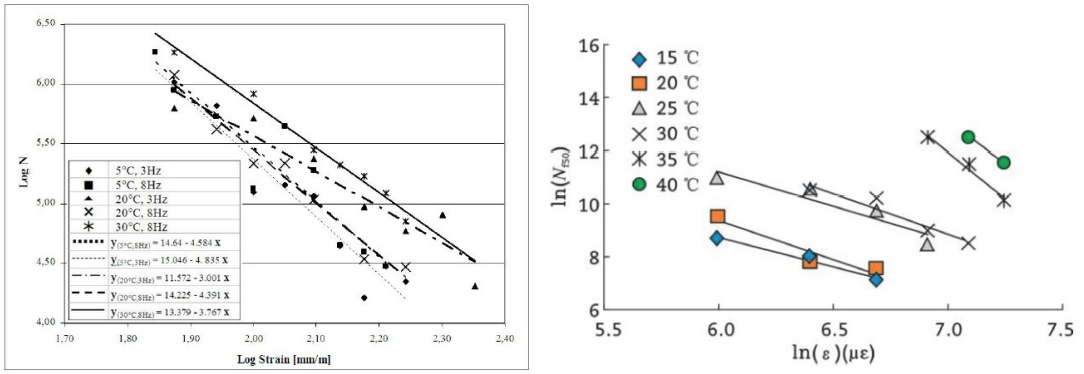
$$N_f = k_1 \epsilon^{k_2} \tag{3.11}$$

$$\log N_f = \log k_1 + k_2 \log \epsilon \tag{3.12}$$

The fatigue line is also plotted as shown in figure 4.1a for a frequency of 30 Hz and using three different strain levels $[\frac{\mu meter}{meter}]$ with six repetitions and then plotting linear regression curve after taking logarithmic values. Figure 4.1a also contains the needed parameters for the fatigue line equation (3.12). This fatigue line is plotted after doing four four-point bending tests by Dura Vermeer on 18 specimens without any sensor attached. Figure 4.1a shows SN curve in logarithmic scale so both the x and y axes are in log scale. X axis is strain values in $\frac{\mu meter}{meter}$ and the y axis is loading cycles which is just the number of times loading cycles can act on the specimen until its stiffness becomes half of the initial stiffness or in other words when fatigue life is reached.

Using the SN curve, fatigue life can be predicted for any strain level by simply reading the y-axis that matches that strain level or simply using the equation (3.12) with the parameters as shown in the figure 4.1a on the right side table.

In this research, asphalt concrete is used as a case study. After training the ML models using sensor data after doing lots of four-point bending tests on Teflon beam which have similar stiffness to the asphalt specimen we can first train the ML models. Using these ML modes we can get strain based on voltage and frequency from a four-point bending test on asphalt beams. However, it should be noted that laboratory tests were limited to only a frequency of 30 Hz due to the limitation of resources while different voltages have been registered based on different strain levels. Section 3.1.3 goes into more in detail on the four-point bending test and how these strain levels are calculated. It should be noted that changing temperature, humidity, and frequency can change the fatigue line curve. However, the master curve (which is also plotted while doing a four-point bending test to get a fatigue line) can be used to solve this issue as different frequency gives different stiffness which can then be used in the phenomenological models or using models that is not sensitive to this can be used for fatigue life prediction. Another solution is to investigate and get different fatigue line curves and use artificial intelligence (machine learning) to learn each data from those curves and then be able to interpolate and extrapolate based on these trained data. Due to the limitation of resources and time, this has not been further investigated. However, doing so in machine learning is not difficult but since getting those SN curves was not possible as fabrication and testing takes a lot of time, it is thus not done in this study. It should be again noted that the purpose of this research is to show the benefits of using a piezoelectric sensor under asphalt concrete beams and using its data (voltage+frequency converted to strain using ML models) in the fatigue life monitoring models to predict fatigue life



(a) SN-curve different temperature and frequency [7]

(b) SN-curve different temperature [65]

Figure 3.32: SN curve related to different temperatures and frequencies

3.5.2. Using Phenomenological models

Phenomenological models used in this research:

- Cheng model
- Yingjun Mei et al.
- Asphalt Institute model

Models that enable or can be used as cumulative damage are useful when service loadings are variable in nature [251].

This research shows great potential for predicting asphalt concrete fatigue life using Miner's rule however, previously has been shown that there is poor match between Miner's damage ratio and observed amount of cracking [7]. However, this can be taken care of by using calibration factors [7].

3.5.2.1. Cheng model

As demonstrated earlier in section 2.5.3, the temperature has a significant impact on determining the fatigue life of asphalt mixtures. It has been substantiated that Equation (2.33) is a practical approach to quantify the influence of temperature on fatigue life, irrespective of the type of mixture and test conducted [65].

In equation (2.33), the fatigue life of the asphalt mixture N_f is influenced by the strain level ϵ_t, the temperature (T), and the fitting parameters A, B, and C. This equation provides a practical means to characterize the impact of temperature on fatigue life, regardless of the specific mixture type or test conducted.

In 2022, Cheng made further improvements to Equation (2.33) by enhancing its ability to account for rest periods and incorporating both temperature and strain levels. These advancements allow for a more comprehensive and accurate characterization of the influence of various factors on fatigue life in asphalt mixtures. This equation as proposed by Cheng in his PhD is also illustrated in equation (2.34)

In Cheng's improved version of Equation (2.33) [5], the fatigue life of the asphalt mixture (N_f) is determined by the strain level (ϵ_t), temperature (T), and rest period (RP), along with the fitting parameters A, B, C, and D. This enhanced equation (2.34) takes into account the combined effects of temperature, strain level, and rest periods to provide a more comprehensive understanding of fatigue life in asphalt mixtures. In this research, the Broyden-Fletcher-Goldfarb-Shanno (L-BFGS-B) algorithm is used to find the fitting parameters.

3.5.2.2. Yingjun Mei et al. model

In this study done by Yingjun et. al. [252], several key findings have emerged. Firstly, the strain fatigue equation for asphalt mixtures was established, revealing that it exhibits minimal temperature dependency. Surprisingly, a single strain equation was found to effectively describe strain fatigue across a wide range of temperature variations, yielding a remarkable correlation coefficient of 92%.

Secondly, a significant regional influence on the fatigue life of asphalt pavement was uncovered. In regions with abundant rainfall, the pavement's fatigue life was observed to be approximately 60% that of arid climates with minimal precipitation. This relationship was further linked to the thickness of the undersurface layer, with the proportionality between rainy and dry climates diminishing as this layer's thickness increased.

Furthermore, it was discovered that augmenting the thickness of the surface layer in asphalt pavement led to a substantial improvement in the pavement's fatigue life, highlighting the practical significance of this design parameter.

The study employed a comprehensive approach to predict the fatigue life of asphalt pavement, involving several detailed steps. Initially, a strain fatigue equation for the undersurface layer of asphalt concrete was established. Next, the cumulative number of standard axle loads within the pavement's design life was determined and categorized into temperature ranges. The maximum tensile strain was then calculated for each temperature range. Subsequently, the predicted numbers of maximum standard axle loads were obtained for each temperature range by integrating the maximum tensile strain into the strain fatigue equation. The fatigue damage degree, accounting for a safety coefficient, was calculated for each temperature range and summed to yield the overall fatigue damage degree of the asphalt pavement. Finally, using the fatigue damage degree and the cumulative number of standard axle loads for each temperature range, the fatigue life of the pavement was predicted.

A detailed explanation of this method is discussed in 2.4.2.1 with the final equation given in equation (2.50).

These equations form the foundation of a comprehensive methodology for predicting the fatigue life of asphalt pavement, providing valuable insights into the complex interplay between temperature, rainfall, and pavement design.

3.5.2.3. Asphalt Institute model

As also mentioned earlier in chapter 2 section 2.4.2 where phenomenological models were discussed it also includes the well-known model proposed by the Asphalt Institute after extensive research and experimentation[66, 67, 68, 69] in equation (2.35). Fitting parameters k_1, k_2 and k_3 are found by using the curve fitting method.

It should be noted that in this research only these models and the SN curve were selected as the goal of this research is not to use every single fatigue life model but to show the potential of how Artificial Intelligence (Machine Learning) can help in automation alongside with using piezoelectric sensors for determining fatigue life of asphalt.

3.6. Fatigue life prediction (damage monitoring models)

In this section, a few monitoring models that have been used to construct damage curves are given. These models are simple and yet can give an indication of damage in asphalt.

3.6.1. Miner's based model(s)

Miner's rule is the most simplest and effective model. It can give an indication of damage and since beams are tested using a 4PB test setup with constant strain, the miner's rule gives a linear line. This method is widely for the range of materials used because it is very simple and easy to apply.

Miner's rule [75], developed around 1945, assumes a linear accumulation of fatigue damage with loading cycles, as shown in Equation (2.41) and illustrated in figure 2.22. Despite its wide adoption due to its computational simplicity, it has been observed that linearity only applies during the intermediate stage of the test. Fatigue damage follows a nonlinear pattern during the initial and final stages of loading repetition.

If damage based on the miner's hypothesis are calculated and drawn against a number of cycles it will be a straight line as shown in figure 2.22.

The curve appears as a straight line because the contribution to damage from each load cycle is consistent. However, it is important to note that the actual formation and growth of cracks may have a different lifespan compared to what is predicted by Miner's criteria.

Hopman et al. (1989) [76]introduced a nonlinear model based on Miner's rule that incorporates an exponent, as depicted in Equation (2.42). Another nonlinear fatigue damage law, described by Equation 21, incorporates linear functions of strain amplitude.

In this context, "D" represents the fatigue damage caused by repetitive loading, "N" represents

the number of loading cycles, and "N_f" represents the fatigue life of the mixture. The parameter "x" is an exponent that falls within the range of 0.82 to 0.92.

3.6.2. Chaboche and Lesne model

Another model used is Chaboche and Lesne [80] which uses a nonlinear model and fitting parameters as shown in equation (2.52).

where D is fatigue damage, N is the loading cycle, N_f is the fatigue life of the mixture,α, β, C are fitting parameters.

3.6.3. Bodin et al. model

Bodin [81] proposed a damage model as shown in the equation (2.53) where parameters are identified using uniaxial fatigue tests. These parameters are estimated in this study using the SN-curve. This model shows promising results to be used in such a way.

3.6.4. Ma et al.

Ma et al. in 2019 [74] proposed model, as illustrated in Equation (2.40). These types of models are particularly valuable as they consider the damage history and current damage state of the mixture, thereby providing a more comprehensive assessment of fatigue life.

In the fatigue life models developed by Ma et al. (2019), the fatigue life of the asphalt mixture N_f is determined by the strain level (ϵ_t), the damage of the mixture D, and the fitting parameters A, B, and C. These models take into account the influence of the damage history and current damage state of the asphalt mixture on its fatigue life. Similar models are discussed more in chapter two 2.4 of this thesis

3.7. Conclusion

This chapter culminates in the presentation of a novel methodology and protocols for leveraging piezoelectric sensors. After detailing the process of creating asphalt beams, the spotlight is directed towards the four-point bending test. The four-point bending (4PB) test utilizes a LVDT to measure deformation, which is then converted into tensile strain using Equation (3.2). This approach enabled to employ Teflon beam with stiffness similar to asphalt beams, generating a substantial amount of sensor data for training machine learning (ML) models. Subsequently, asphalt beams equipped with PZT sensors was tested in 4PB test to predict their fatigue life, based on strain data converted from sensor readings using pre-trained ML models. To validate the observed patterns from the Teflon beam 4PB test, COMSOL Finite Element Analysis (FEA) was utilized for numerical analysis, and a consistent pattern and trendline were verified.

In this chapter, an in-depth explanation of the ML pipeline is provided, encompassing each step from data preprocessing to hyperparameter tuning. Towards the chapter's conclusion, a collection of fatigue life prediction models applicable to this method for predicting the fatigue life of asphalt is presented. Furthermore, models used for calculating and visualizing damage in asphalt beams, relevant to this methodology, are discussed.

4

Results

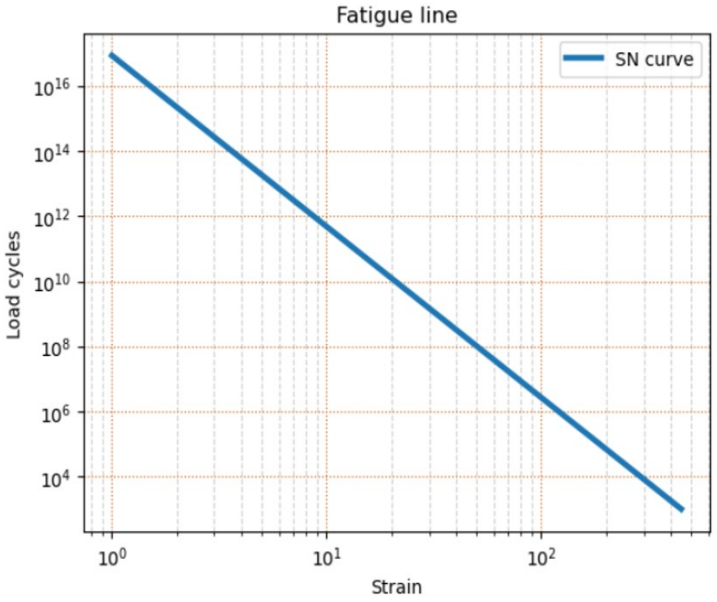
important to note that the plots generated from sensor data have undergone postprocessing using Python code, which is provided in the appendices (C and C). The postprocessing of the voltage-time curves involves several steps, including the plotting of results for voltage amplitude and saving them so that they can be used for machine learning. This focus on voltage amplitudes is particularly relevant to this research and serves to streamline the data processing by reducing the dataset's size.

Furthermore, the postprocessing also involves aligning all the voltage-time curves. This alignment enhances readability and contributes to a visually appealing presentation of the data, while ensuring that the critical information is conveyed as effectively as possible.

4.1. Four-point bending test result

This section encompasses the results obtained from the four-point bending test. The initial four-point bending test was conducted at the Dura Vermeer laboratory, where PZT sensors were not utilized. These results are presented in Figure 3.12 for the S-N curve and in Figure 4.2 for the master curve.

Following this, the results from the four-point bending test performed at the Tu Delft library are showcased. Initially, the section presents all the results obtained from the PZT sensor when a Teflon beam with a PZT sensor was employed in the four-point bending test. Subsequently, in Section 4.3.1, after confirming a similar pattern (voltage amplitude vs strain curve) through COMSOL analysis, the results of asphalt beams are compared with those of the Teflon beam.



(a) SN curve

Fatigue line $N = k_1 \cdot \varepsilon^{k2}$	
$\log_{10} k_1$:	**16.934**
k_2:	**-5.253**

Characteristics	
Number of specimen:	18
Cycles for strain-charact.:	1000000
Mean value of E_{ini} [MPa]:	10874
strain-characteristics [µm/m]:	**120.6**

(b) SN curve equation parameters

Figure 4.1: SN Curves and Parameters

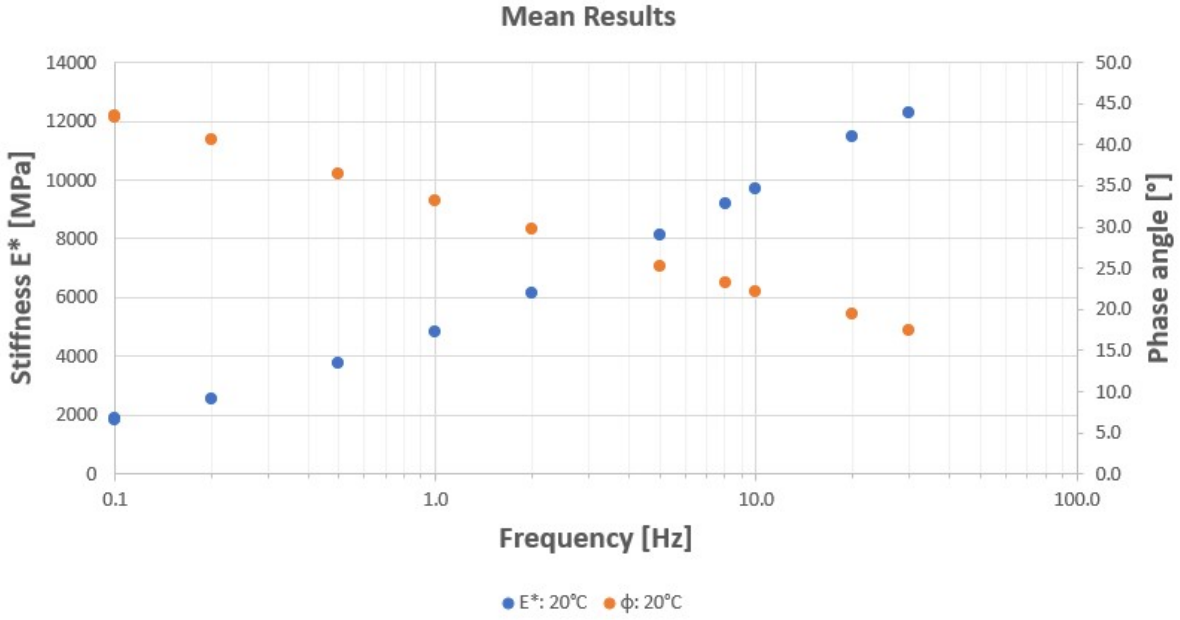


Figure 4.2: Master curve

4.1.1. PZT sensor output data

As mentioned in section 3.1.2 and 3.1.3. it is understood how the sensor will be attached and how four-point bending tests are done. The figures below illustrate the results of sensor data if the frequency is set constant and strain values are changed.

The point-bending test program is again summarized in the table 4.1.

Pulse Width (ms)	Frequency [Hz]	Strain [μmeter/meter]
1000	1	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
500	2	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
400	2.5	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
200	5	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
100	10	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
66.67	15	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
50	20	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
40	25	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
33.33	30	15, 50, 100, 150, 200, 250, 300, 350, 400, 450
28.57	35	15, 50, 100, 150, 200, 250, 300, 350, 400, 450

Table 4.1: Four-point bending test program

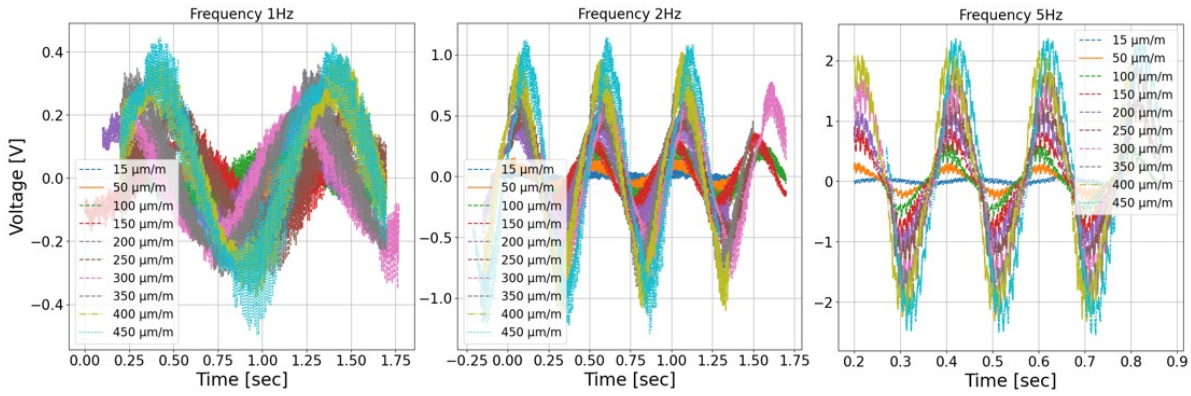


Figure 4.3: PZT sensor output: Voltage vs time plot for loading frequency 1 [Hz], 2[Hz] and 5[Hz]

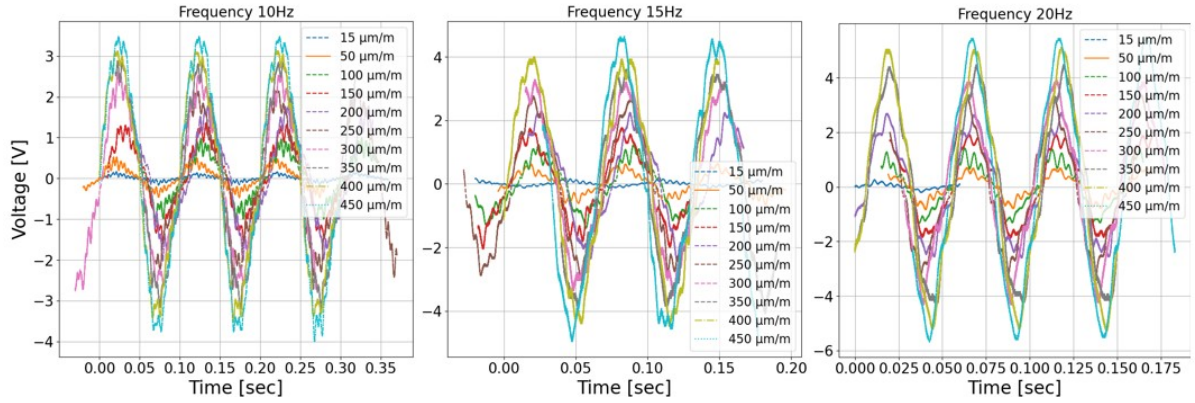


Figure 4.4: PZT sensor output: Voltage vs time plot for loading frequency 10 [Hz], 15 [Hz], and 20 [Hz]

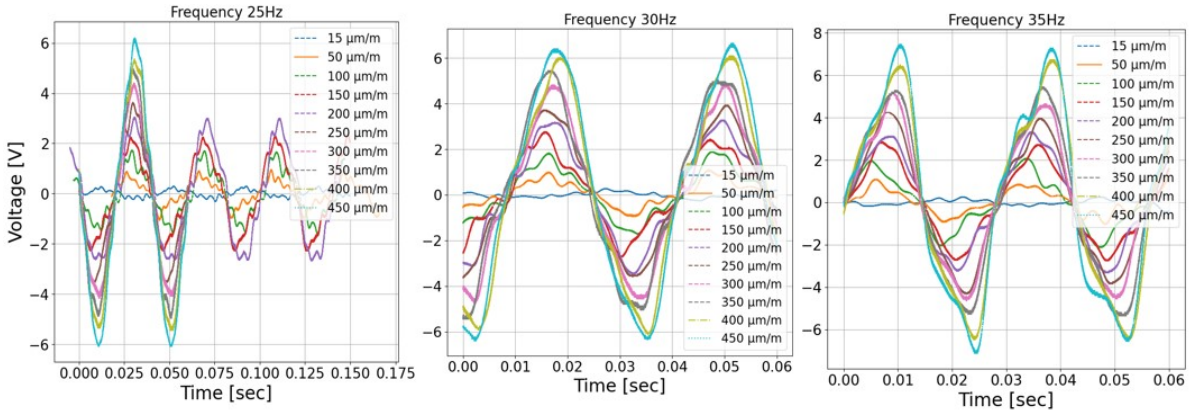


Figure 4.5: PZT sensor output: Voltage vs time plot for loading frequency 25 [Hz], 30 [Hz], and 35 [Hz]

It can be concluded that as strain level increases so is the voltage. This is in line with theory as seen from the equations in Chapter 2 section 2.2.0.1. More strain just bends the sensor more which activates more piezoelectric effect. Also, the sensor output can be plotted if strain is kept constant and frequency is varied.

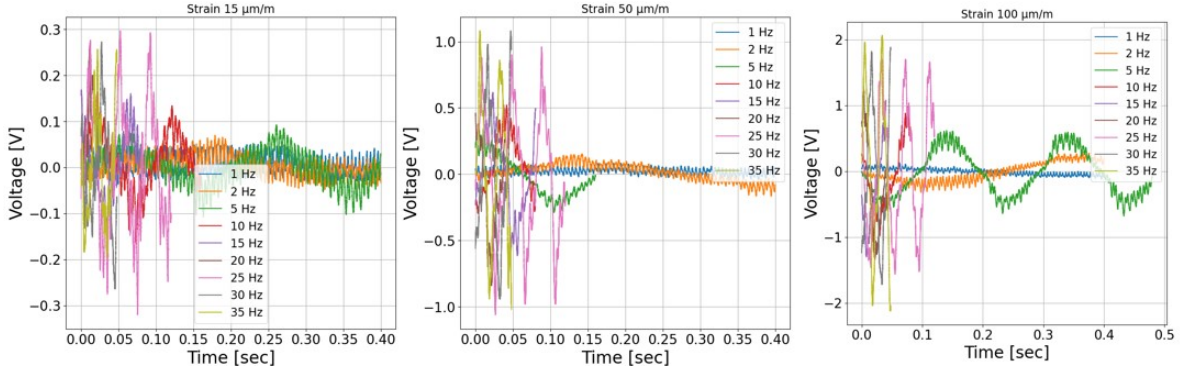


Figure 4.6: PZT sensor output: Voltage vs time plot for strain 15 $[\frac{\mu m}{m}]$, 50 $[\frac{\mu m}{m}]$ and 100 $[\frac{\mu m}{m}]$

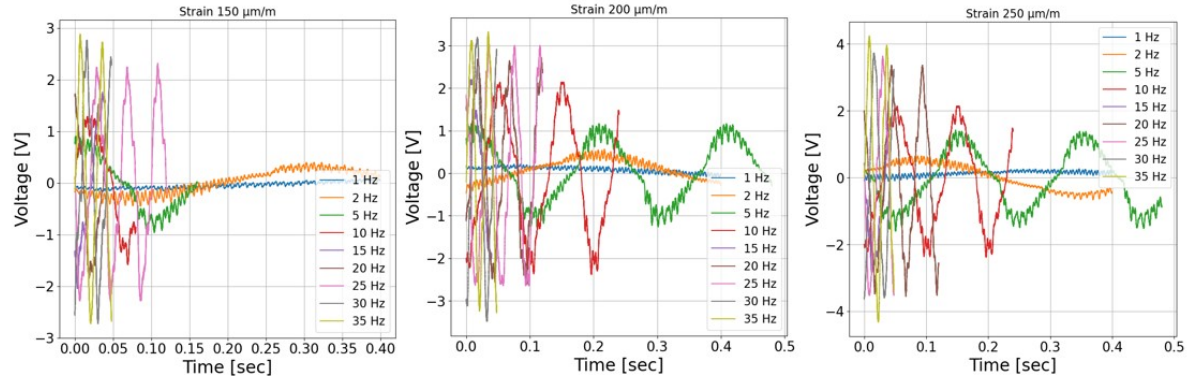


Figure 4.7: PZT sensor output: Voltage vs time plot for strain 150 $[\frac{\mu m}{m}]$, 200 $[\frac{\mu m}{m}]$ and 250 $[\frac{\mu m}{m}]$

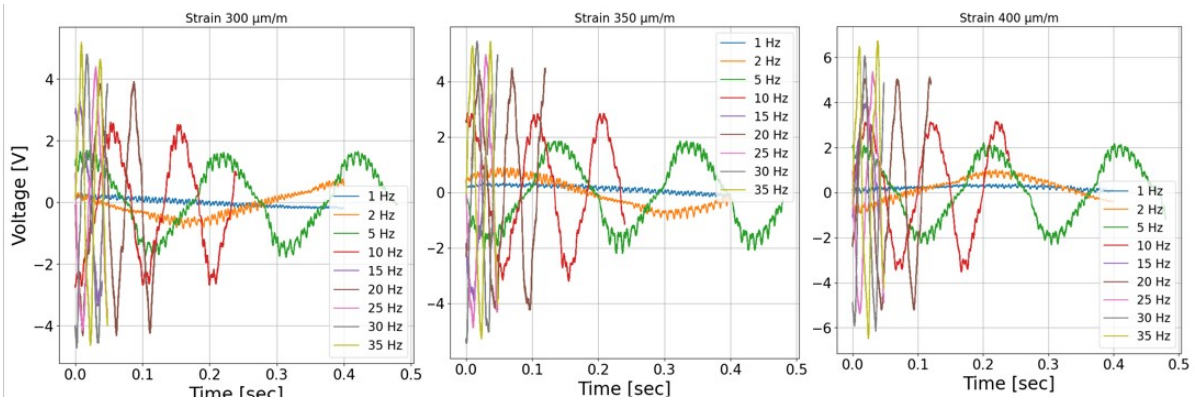


Figure 4.8: PZT sensor output: Voltage vs time plot for strain 300 $[\frac{\mu m}{m}]$, 350 $[\frac{\mu m}{m}]$ and 400 $[\frac{\mu m}{m}]$

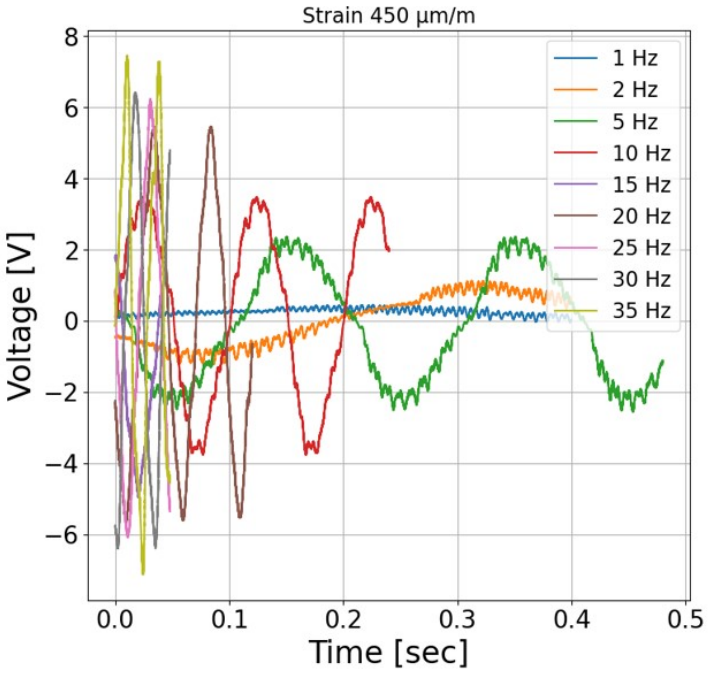
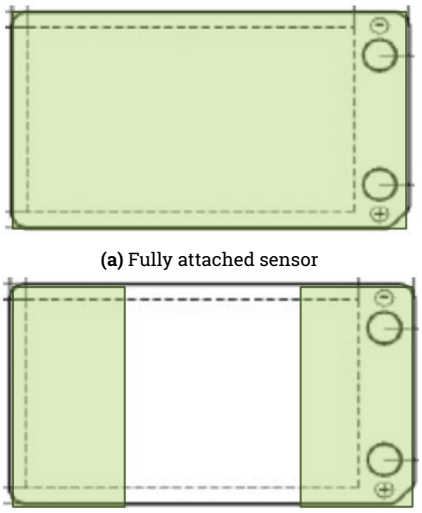


Figure 4.9: PZT sensor output: Voltage vs time plot for strain 450 $[\frac{\mu m}{m}]$

It can be seen from the graphs above that as the frequency increases, the peak-to-peak voltage also increases. A similar pattern is also seen as the strain values were increased, and voltage was also increased. These data are collected and a general plot such as voltage vs strain plot and voltage vs frequency plot are plotted in section 4.4.1. Also, **integral voltage vs strain** is plotted which is included in Appendix D.

4.1.2. Influence of sensor-substrate attachment

During the experimental phase, it is found that attaching the sensor to the substrate can create a big difference in the results. Just changing the way the sensor is attached to the substrate can affect results greatly. The table 4.2 and 4.3 shows the difference between the results if the attachment is fully attached to the substrate as shown in figure 4.10a and partially attached as shown in figure 4.10b.



(a) Fully attached sensor

(b) Partially attached sensor only at the ends

Figure 4.10: Sensor attachment techniques with the adhesive area highlighted

Table 4.2: Voltage amplitude from PZT sensor - Sensor fully Attached

Strain (µm/m)	Voltage amplitude [V]							
	1	**2**	**5**	**10**	**15**	**20**	**25**	**30**
15	0.086	0.25	0.266	0.384	0.581	0.646	0.763	0.746
50	0.123	0.37	0.66	1.128	1.715	1.53	1.89	2.125
100	0.223	0.74	1.228	2.05	2.555	2.78	3.23	3.65
150	0.243	1.294	2.275	2.91	3.565	3.97	4.62	5.11
200	0.541	1.5	2.595	3.645	4.49	5.01	5.85	5.43
250	0.613	1.5	2.86	4.375	5.43	6.31	7.15	7.72

Table 4.3: Voltage amplitude from PZT sensor - Sensor partially Attached

Strain (µm/m)	Voltage amplitude [V]							
	1	**2**	**5**	**10**	**15**	**20**	**25**	**30**
15	0.088	0.096	0.098	0.096	0.099	0.1	0.092	0.11
50	0.086	0.097	0.095	0.099	0.098	0.111	0.122	0.127
100	0.089	0.094	0.092	0.096	0.114	0.12	0.138	0.124
150	0.09	0.095	0.097	0.108	0.125	0.122	0.141	0.136
200	0.102	0.094	0.109	0.098	0.129	0.138	0.15	0.151
250	0.098	0.149	0.16	0.16	0.13	0.149	0.16	0.16

A fully attached sensor makes sure that the sensor is fully activated which in return generates good voltage. Partially attaching sensors do not generate any voltage. These results only be concluded if it is assumed or known that everything else was running fine and there was no other influence from the oscilloscope or the probes used during the experiment. Since there were no other reasons for this difference, it has therefore concluded that the only reason is the attachment difference.

4.2. Simulation results

4.2.1. Simulated piezoelectric voltage

Appendix F includes all the contour plots from the sensor based on different strain levels. This section presents results in the form of a table which includes voltage amplitudes from COMSOL FEA, voltage amplitude from experiment, and percentage error difference between them for different strain levels. Percent error is only calculated for frequency of 20 [Hz] and 30 [Hz] as at

30 [Hz] and strain level of 200 $[\frac{\mu m}{m}]$ data recorded from the experiment was not accurate due to machine transducer defect.

Table 4.4: Voltage amplitude from 4PB test and COMSOL FEA program for a frequency of 1 [Hz]

Strain [-]	4PB-test [V]	COMSOL [V]
15	0.2655	0.24
50	0.66	0.65
100	1.228	1.3
200	2.595	2.52
300	3.275	3.57
400	4.17	4.42

Table 4.5: Voltage amplitude from 4PB test and COMSOL FEA program for a frequency of 10[Hz]

Strain [-]	4PB-test [V]	COMSOL [V]
15	0.384	0.34
50	1.128	1.08
100	2.05	2.08
200	3.645	4.05
300	5.13	5.3
400	6.34	6.5

Table 4.6: Voltage amplitude from 4PB test and COMSOL FEA program for a frequency of 20[Hz]

Strain [-]	4PB-test [V]	COMSOL [V]	% difference
15	0.646	0.65	0.619195
50	1.53	1.5	-1.96079
100	2.78	2.75	-1.07914
200	5.01	5.24	4.59082
300	7.59	7.75	2.10804
400	9.52	9.6	0.840336

Table 4.7: Voltage amplitude from 4PB test and COMSOL FEA program for a frequency of 30[Hz]

Strain [-]	4PB-test [V]	COMSOL [V]	% difference
15	0.746	0.74	-0.80429
50	2.125	2.1	-1.17647
100	3.65	3.71	1.64384
200	5.43	5.32	-2.02578
300	8.92	8.71	-2.35426
400	11.52	11.82	2.60417

4.2.2. Simulated beam deflection

Sinusoidal load is applied on Teflon beams with PZT sensor and the maximum beam deflection in the middle of the beam for different strain levels is attached in Appendix G. Deflection as a result of bending can be summarized in table 4.4 where strain is calculated using this deflection (δ) in equation (3.2). This deflection is the same as observed during the 4PB and COMSOL simulation. Using deflection, ϵ_t can be calculated using equation (3.2) since the height of specimen (h) is 50 mm and also the distance between the support clamp and loading clamp remains the same and constant as given in NEN standards.

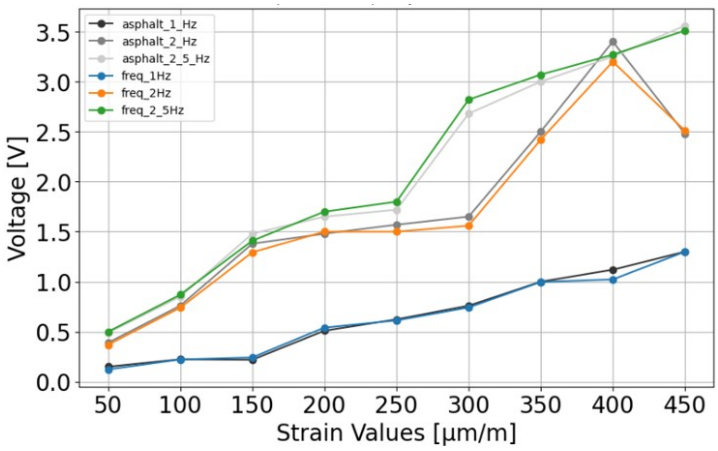
Table 4.8: deflection from 4PB machine and COMSOL FEA program

ϵ [-]	δ_{COMSOL} [mm]	δ_{4PB} [mm]
15	0.008	0.0081
50	0.026	0.030
100	0.053	0.060
200	0.120	0.110
300	0.170	0.162
400	0.220	0.218

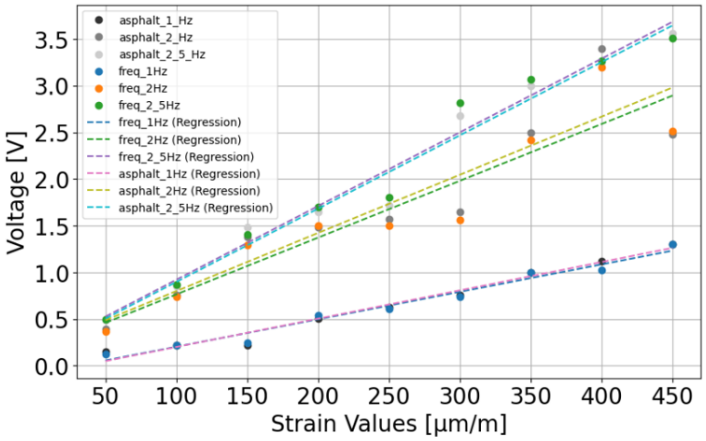
4.3. Comparison of signal processing results

4.3.1. Piezoelectric voltage from four-point bending experiments

In the figure 4.11a and figure 4.11b, a comparison of models has been shown. Figure 4.11a shows a comparison between the results of Teflon beam and Asphalt beams tested under four four-point bending tests at a frequency of 1 Hz, 2 Hz, and 2.5 Hz. Figure 4.11b shows linear regression based on these data.



(a) Comparison between Teflon beams and Asphalt beams under four-point bending



(b) Comparison between Teflon beams and Asphalt beams under four-point bending

Figure 4.11: Comparison of voltage amplitude between Teflon beams and Asphalt beams

The results obtained from both materials exhibit a consistent pattern and trend line, aligning with the theoretical framework outlined in Section 3.1.3. However, it is advisable to conduct additional repetitions and tests, encompassing a broader range of strain levels and frequencies,

to investigate the material properties in even greater detail.

Nevertheless, taking into consideration the similarity in behavior between the asphalt beam and Teflon beam, and the fact that the machine applies varying forces to maintain the same deformation (which is then converted to strain according to Equation (3.2)), if strain levels of 220 $[\frac{\mu m}{m}]$, 160 $[\frac{\mu m}{m}]$, and 114 $[\frac{\mu m}{m}]$ are applied to the asphalt beam with a frequency of 30 [Hz], the resulting voltage amplitude from the PZT sensor are 6.67 [V], 5.09 [V], and 3.87 [V], respectively.

Utilizing these voltage and frequency combinations within the ML-models yields the following results, as summarized in Table 4.9.

Table 4.9: Comparison of Different Regression Models

Model	Strain for PZT Voltage			Average % Diff
	6.67[V]	5.09[V]	3.87[V]	
Real strain	**220**	**160**	**114**	**0**
Linear Regression	201	131	76	-21.9
Ridge Regression	202	131	77	-19.7
Lasso Regression	202	132	78	-18.8
ElasticNet Regression	202	132	78	-19.1
Polynomial Regression	201	143	100	-10.3
SVR	230	148	50	-19.6
K-Nearest Neighbors	227	153	172	16.6
Random Forest	228	169	146	12.5
Decision Tree	200	150	150	5.4
Gradient Boosting Regression	**206**	**156**	**138**	**4.1**
Extra Trees Regression	**221**	**151.5**	**132.5**	**3.9**
Neural Network	214	152	102	-6.0
Bagging Regression	228	164	144	10.9
CatBoost Regression	**218**	**159**	**115**	**0.7**

Table 4.9 provides a clear indication that the CatBoost Regression, ExtraTrees Regression and Gradient Boosting Regression did well when predicting strain based on voltage amplitude (6.67 [V], 5.09 [V], and 3.87 [V]) and frequency of 30[Hz] combination. However, when these ML models were subsequently compared with each other, Table 4.14 demonstrated that the CatBoost Regression, Bagging Regression, Neural Network, and ExtraTrees Regression consistently exhibited strong performance in terms of their average accuracy. Its essentail to point that if ML model performs better at this stage do not necessarily mean that they perform well in average or in general. Table 4.14 instead deals with general accuracy of the models. It is also observed that as strain level increases or frequency, ML models and PZT sensor perform better.

Another observation is that the difference between voltage from LVDT and the sensor (CH1) gives one value regardless of any frequency as described in the next section. Results from PVDF are ignored here and only attached in the Appendix E.

4.3.2. LVDT and piezoelectric voltage

One of the results from the 4PB test shows that subtracting voltage from the piezoelectric sensor and LVDT gives single data which is plotted with a linear regression curve. Results from the data show that it does not matter what frequency it is used, after calculating absolute difference, only one data is produced.

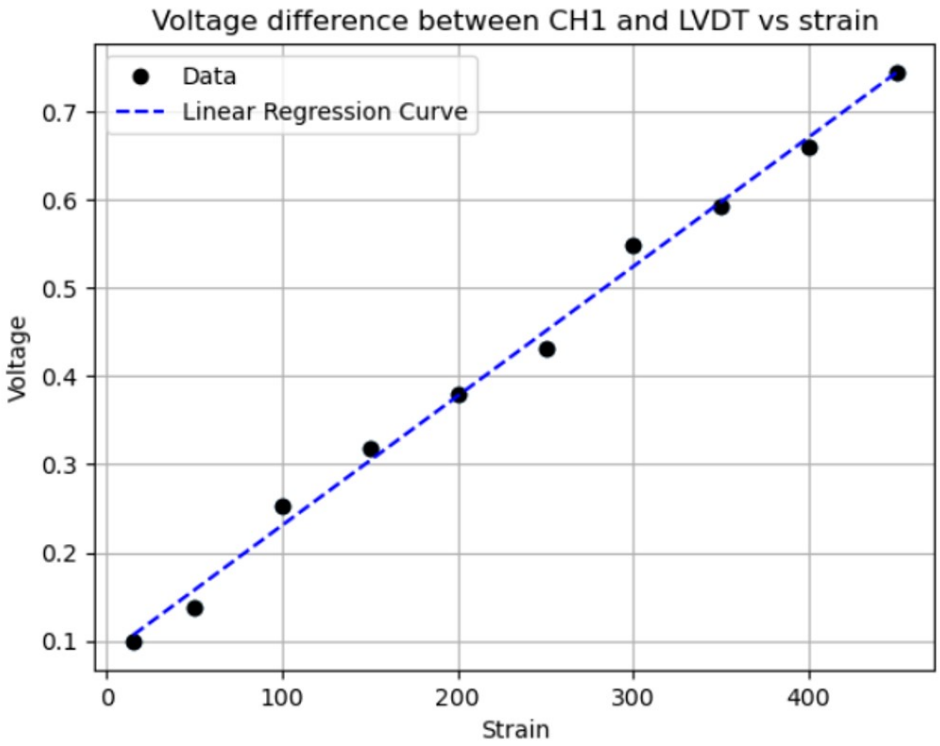


Figure 4.12: Difference between voltage from sensor and LVDT

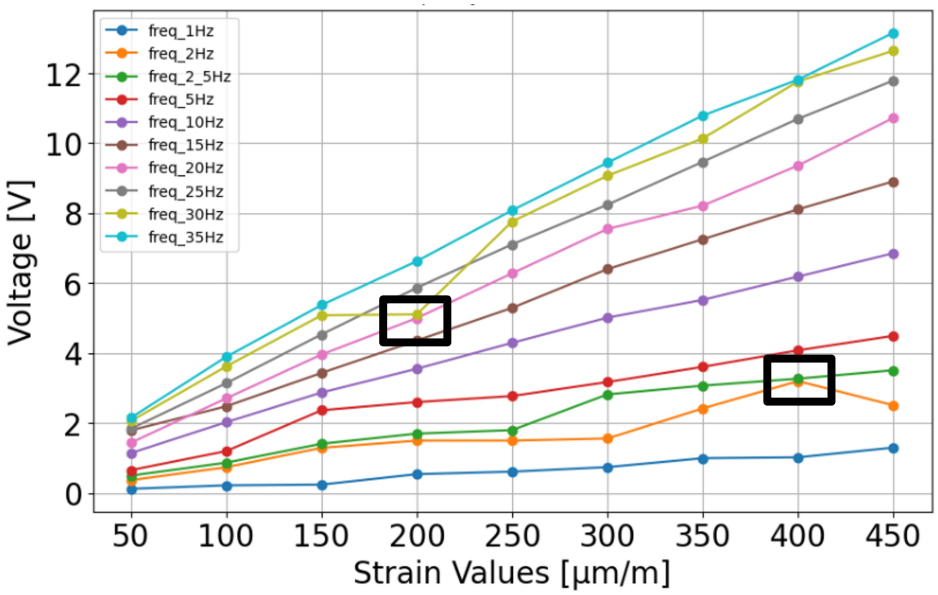
4.4. Machine learning

This section deals with different results relevant to the Machine learning part of the thesis from data processing results to final model prediction.

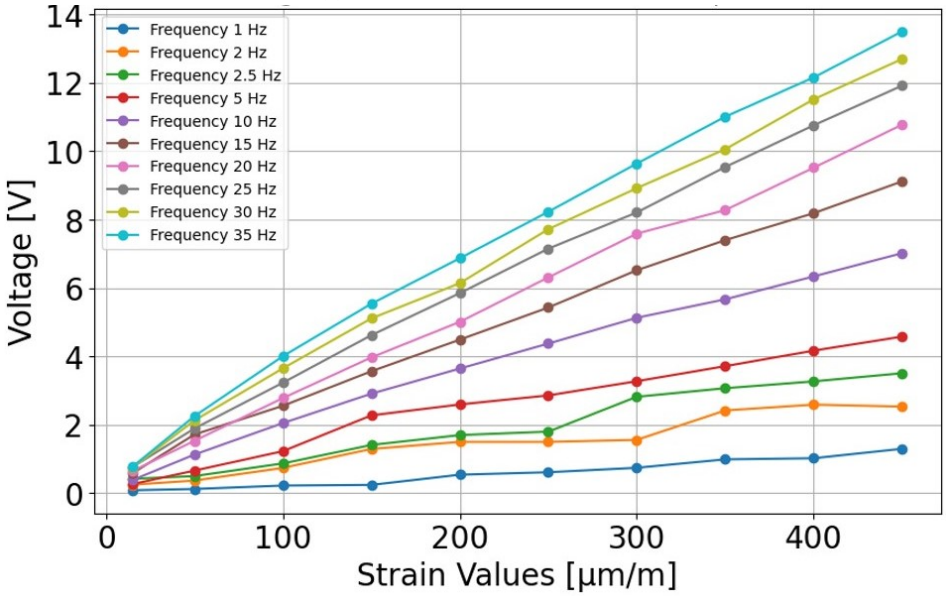
Voltages are registered from the piezoelectric sensor when the Teflon substrate has been used and these values are required for training ML models. After post-processing results in Python voltage(amplitude) vs strain plots can be plotted. These values are voltage amplitudes or peak-to-peak voltage as seen in figures under the section "sensor output data" at the beginning of this chapter.

4.4.1. Results from data preprocessing

In order to use data for ML training, correction of some data must be done as part of feature engineering or data preprocessing of ML pipeline. Voltage for strain 200 $\frac{\mu meter}{meter}$ and 400 $\frac{\mu meter}{meter}$ are corrected by interpolating the data. These errors are experimental errors and are left to show that the purpose of this step in the ML pipeline is to correct such errors. Voltage strain curves after correction are plotted in figure 4.13b.



(a) Voltage strain before correction



(b) Voltage strain after correction

Figure 4.13: Voltage amplitude [V] vs strain $[\frac{\mu m}{m}]$ plots before and after feature engineering

Also, linear regression lines can be plotted for the data as figure 4.14. The trend follows the theory as discussed in chapter 2 "Literature Study" section "Sensor 2.2" as the voltage is directly proportional to the strain. Other relevant figures include the voltage frequency curve (figure 4.15) and 3D plot (4.16) of the voltage-frequency-strain curve where the trend can be clearly seen that **increasing strain or frequency increases voltage amplitude**.

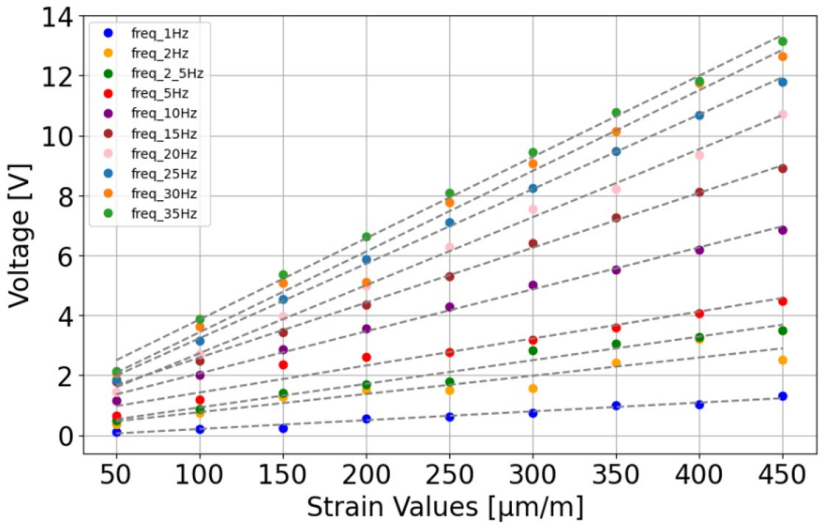
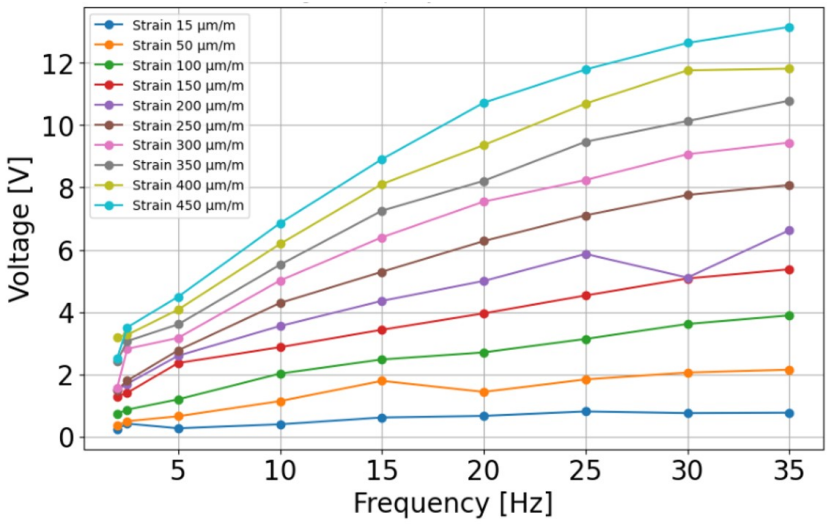
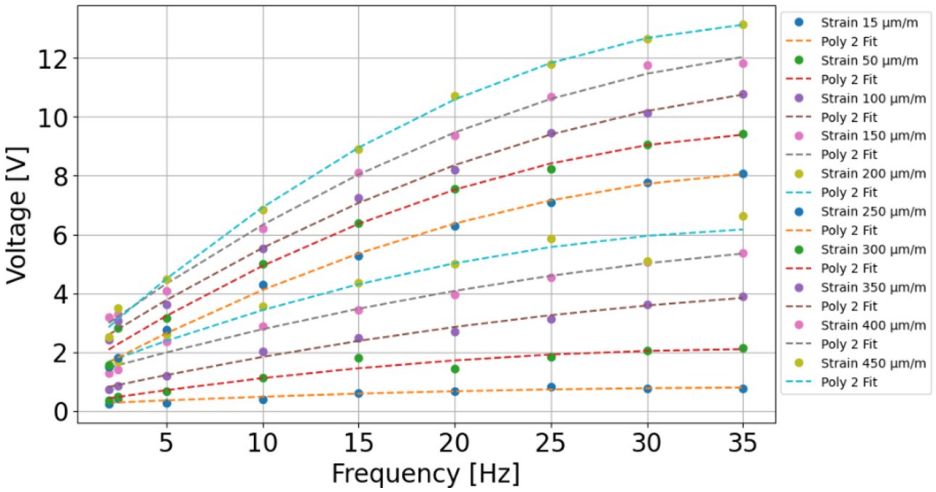


Figure 4.14: Voltage strain linear regression curves



(a) Voltage frequency plot



(b) Voltage frequency polynomial regression plot

Figure 4.15: Voltage amplitude [V] vs frequency [Hz] plot

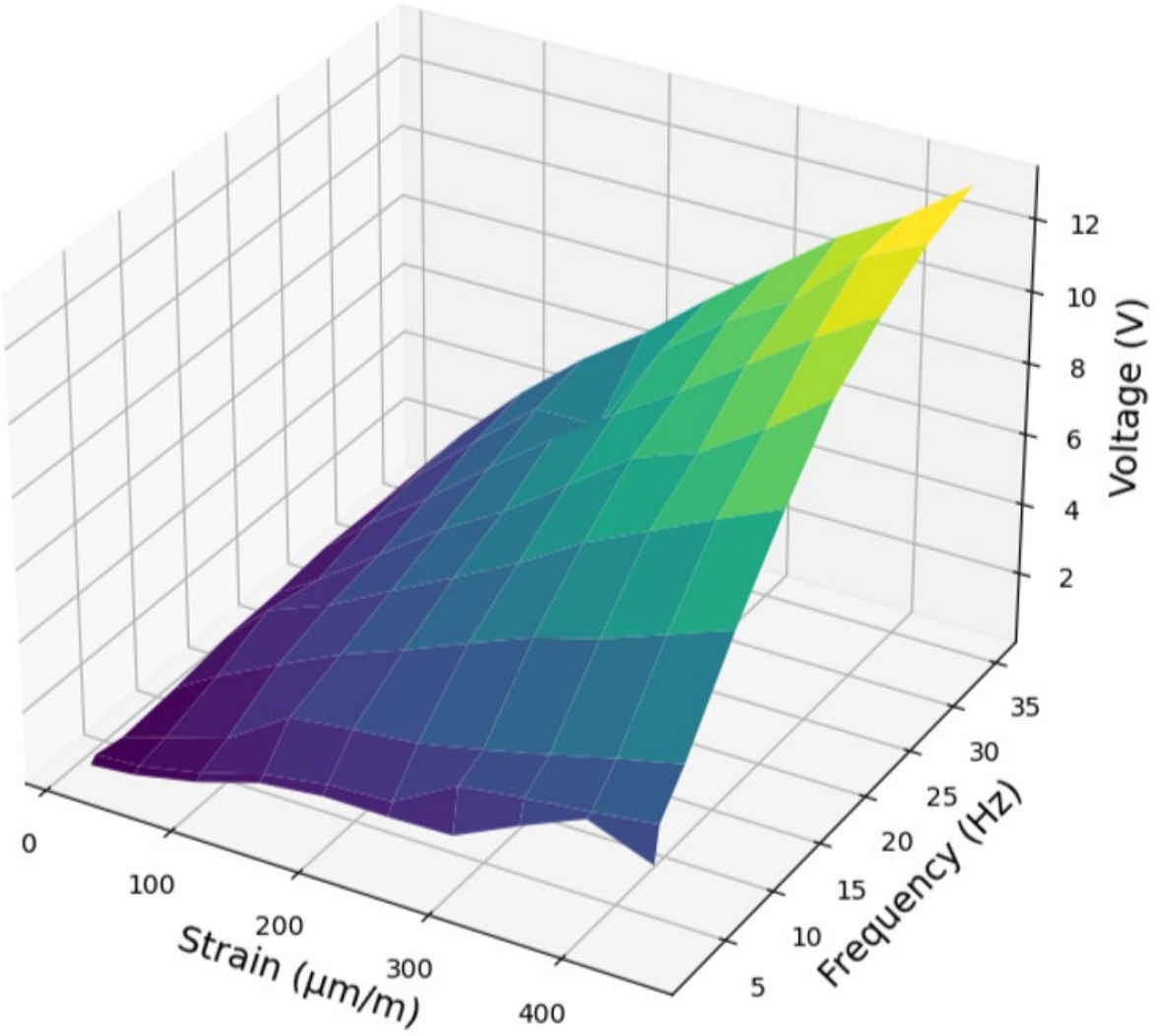


Figure 4.16: 3D surface plot of voltage frequency and strain

4.4.2. ML hyperparameters

One of the most important processes in machine learning is tuning of its hyperparameters as described in detail in chapter 3 "Methodology" section 3.4.5. As described, this is done automatically by the machine with the help of using "Gridsearch" algorithm. Table 4.10 shows the predetermined hyperparameters so that the algorithm can autonomously identify the optimal configuration. Table 4.12 shows the optimal configuration selected by Gridsearch algorithm by minimizing negative root mean squared error (RMSE).

Table 4.10: ML Hyperparameters before tuning

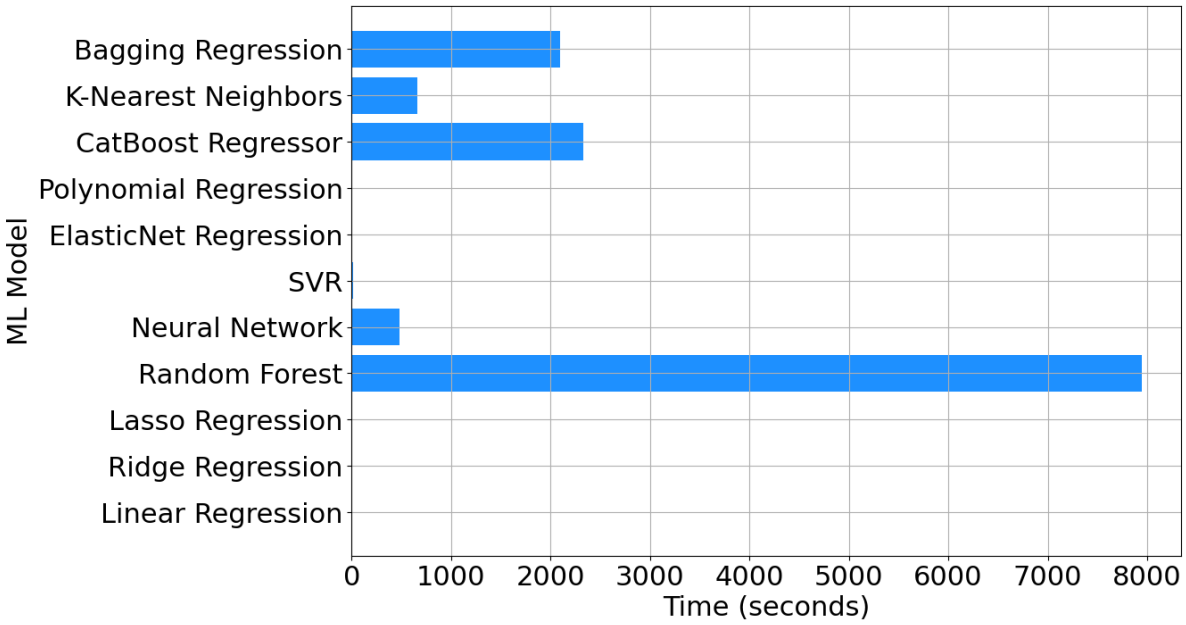
ML Hyperparameters before tuning	
Model	**Hyperparameters**
CatBoost Regressor	'iterations': [2, 4, 6, 8, ...,100] 'learning_rate': [0.01, 0.1, 0.2] 'depth': [6, 8, 10, 15] 'l2_leaf_reg': [1, 3, 5, 7, 9, 10]
SVR	'C': [0.1, 1, 10, 100, 500, 1000, 5000, 10000] 'epsilon': [0.01, 0.1, 0.2, 0.5, 1] 'kernel': ['rbf'] 'degree': [2, 3, 4] 'gamma': [0.1, 1, 2, 5, 10]
ElasticNet Regression	'alpha': [0.01, 0.06, 0.11, 0.16, ...,20] 'l1_ratio': [0.01, 0.06, 0.11, 0.16, ...,20]
Polynomial Regression	'polynomialfeatures__degree': [1, 2, 3, 4, 5]
Linear Regression	
Ridge Regression	'alpha': [0, 0.20202020202020202, 0.40404040404040403, ...,20]
Lasso Regression	'alpha': [0, 0.20202020202020202, 0.40404040404040403, ...,20]
Bagging Regression	'n_estimators': [2, 4, 6, 8, ...,200] 'max_samples': [0.5, 0.55, 0.6, 0.65, ...,1.0] 'max_features': [0.1, 0.5, 0.75, 1.0, 2, 5]
Decision Tree	'criterion': ['squared_error', 'friedman_mse', 'mae'] 'splitter': ['best', 'random'] 'max_depth': [None, 10, 20, 30, 50, 100, 200, 500] 'min_samples_split': [2, 5, 10] 'min_samples_leaf': [1, 2, 4, 8] 'max_features': ['auto', 'sqrt', 'log2', None] 'max_leaf_nodes': [None, 10, 20] 'min_impurity_decrease': [0.0, 0.001, 0.1, 0.2] 'min_weight_fraction_leaf': [0.0, 0.001, 0.1, 0.2]
Gradient Boosting Regression	'n_estimators': [50, 100, 200, 500] 'max_depth': [3, 4, 5]
Extra Trees Regression	'n_estimators': [50, 100, 200] 'max_depth': [None, 10, 20] 'min_samples_split': [2, 5, 10] 'bootstrap': [True, False] 'warm_start': [True, False]
Neural Network	'hidden_layer_sizes': [(), (10,), (10, 10), (10, 10, 10), ...] 'alpha': [0.0001, 0.001, 0.01, 0.1, 1, 2] 'max_iter': [1000]
Random Forest	'n_estimators': [50, 100, 200, 400, 800, 1000, 1200, 1400, 1600, 1800, 2000] 'max_features': ['auto', 'sqrt'] 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None] 'min_samples_split': [2, 5, 10] 'min_samples_leaf': [1, 2, 4, 6] 'bootstrap': [True, False]
K-Nearest Neighbors	'n_neighbors': [1, 2, 3, 4, ...,200] 'weights': ['uniform', 'distance'] 'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'] 'leaf_size': [1, 3, 5, 7, ...,53] 'p': [1, 2]

Table 4.12: ML models and their hyperparameters after tuning.

ML Hyperparameters after tuning	
Model	**Hyperparameters**
Linear Regression	No parameter
Ridge Regression	'alpha': 15.56
Lasso Regression	'alpha': 8.69
ElasticNet Regression	'alpha': 0.27 'l1_ratio': 0.11
Polynomial Regression	'polynomialfeatures__degree': 4
SVR	'C': 5000 'degree': 2 'epsilon': 0.01 'gamma': 0.1 'kernel': 'rbf'
K-Nearest Neighbors	'algorithm': 'auto' 'leaf_size': 5 'metric': 'chebyshev' 'n_neighbors': 2 'p': 1 'weights': 'distance'
Random Forest	'bootstrap': True 'max_depth': None 'max_features': 'auto' 'min_samples_leaf': 1 'min_samples_split': 2 'n_estimators': 100
Decision Tree	'criterion': 'friedman_mse' 'max_depth': 500 'max_features': None 'max_leaf_nodes': None 'min_impurity_decrease': 0.2 'min_samples_leaf': 1 'min_samples_split': 2 'min_weight_fraction_leaf': 0.001 'splitter': 'random'
Gradient Boosting Regression	'max_depth': 3 'n_estimators': 200
Extra Trees Regression	'bootstrap': False 'max_depth': None 'min_samples_split': 2 'n_estimators': 100 'warm_start': True
Neural Network	'alpha': 1 'hidden_layer_sizes': (10, 10, 10, 10, 10) 'max_iter': 1000
Bagging Regression	'max_features': 1.0 'max_samples': 0.94 'n_estimators': 18
CatBoost Regressor	'depth': 6 'iterations': 94 'l2_leaf_reg': 1 'learning_rate': 0.2

4.4.3. ML time

Bar chart in figure 4.17 shows how much time in seconds is required for the model's hyperparameters to be tuned.



(a) ML time required for hyperparameter tuning in bar graph

Model Name	Time (seconds)
Linear Regression	0.176488
Ridge Regression	1.34048
Lasso Regression	1.08946
Random Forest	7946.34
Neural Network	484.682
SVR	18.5677
ElasticNet Regression	2.50168
Polynomial Regression	0.155766
CatBoost Regressor	2325.86
K-Nearest Neighbors	656.994
Bagging Regression	2092.4

(b) Time in seconds in table form

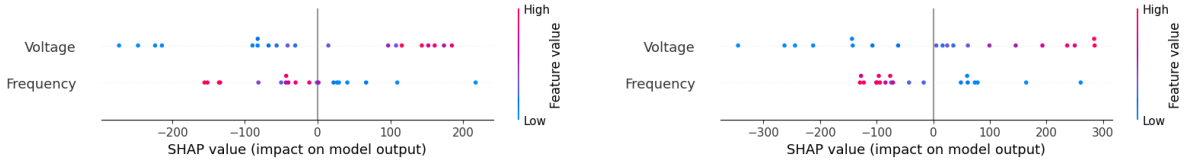
Figure 4.17: Time [sec] required for hyperparameters tuning

The time required for using these models in order to predict strain based on a combination of voltage and frequency is pretty much instant and hence it is not plotted for comparison.

4.4.4. General sensitivity analysis

General sensitivity analysis shows that the voltage is more relevant than frequency which is trivial as there are only two features that have been used for machine learning. Frequency on the other hand does have some relevance as different frequency have their own set of voltages corresponding to certain strain levels. Another observation is the contribution of voltage, it is as positive as negative, or in other words there is balance and a big contribution overall to the

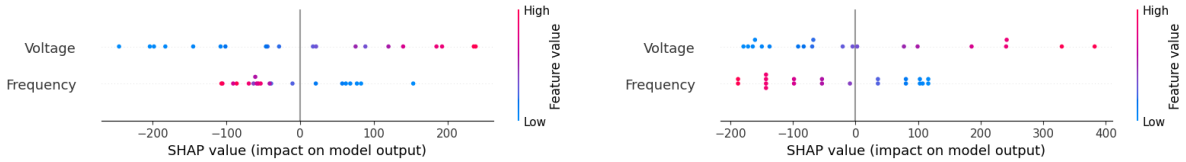
model prediction quality.



(a) SHAP values for SVR Model

(b) SHAP values for Gradient Boosting Model

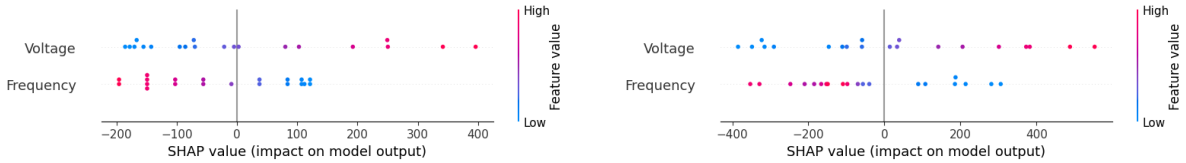
Figure 4.18: Side-by-side SHAP values for SVR and Gradient Boosting regression models.



(a) SHAP values for Bagging Model

(b) SHAP values for Lasso Model

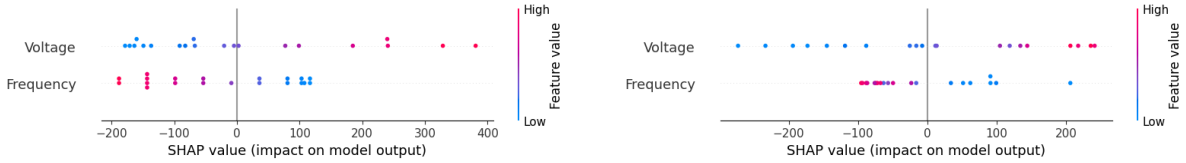
Figure 4.19: Side-by-side SHAP values for Bagging and Lasso models.



(a) SHAP values for Linear Model

(b) SHAP values for Polynomial Model

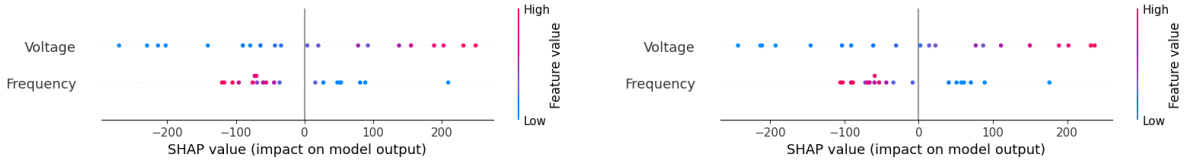
Figure 4.20: Side-by-side SHAP values for Linear and Polynomial models.



(a) SHAP values for ElasticNet Model

(b) SHAP values for Decision-Tree Model

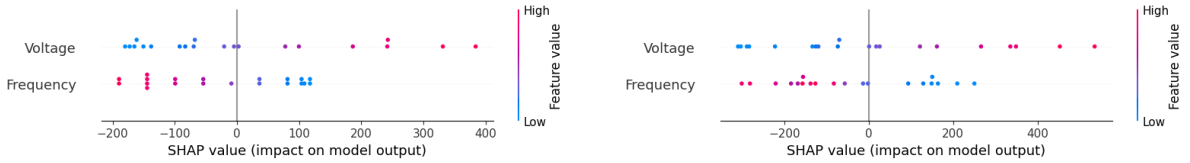
Figure 4.21: Side-by-side SHAP values for ElasticNet and Decision-Tree models.



(a) SHAP values for Extra Trees Model

(b) SHAP values for Random Forest Model

Figure 4.22: Side-by-side SHAP values for Extra Trees and Random Forest models.



(a) SHAP values for Ridge Model

(b) SHAP values for Neural Network Model

Figure 4.23: Side-by-side SHAP values for Ridge and Neural Network models.

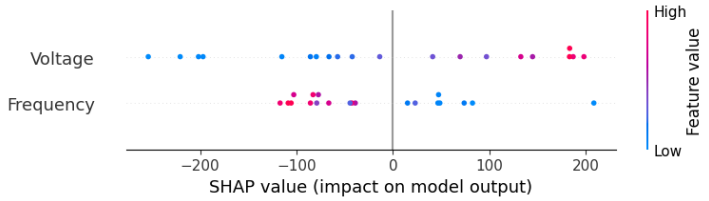
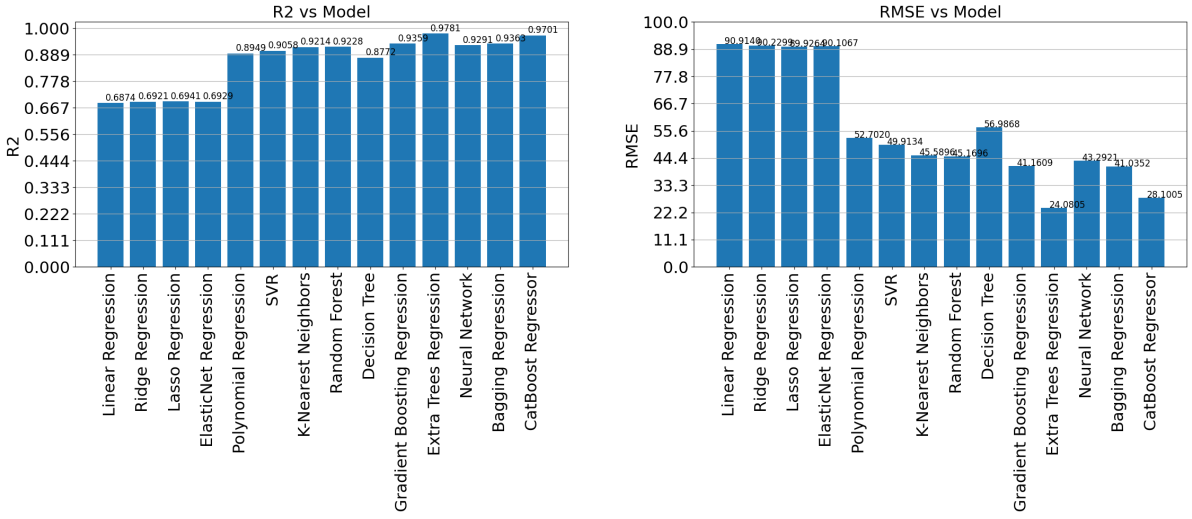


Figure 4.24: SHAP values for K-Nearest Neighbour Model

4.4.5. Comparison of Machine learning models

In this section, ML models are compared based on their prediction capability and how far the prediction is from the actual strain. These are done on the basis of well-known methods as described in section 3.4.5.1.

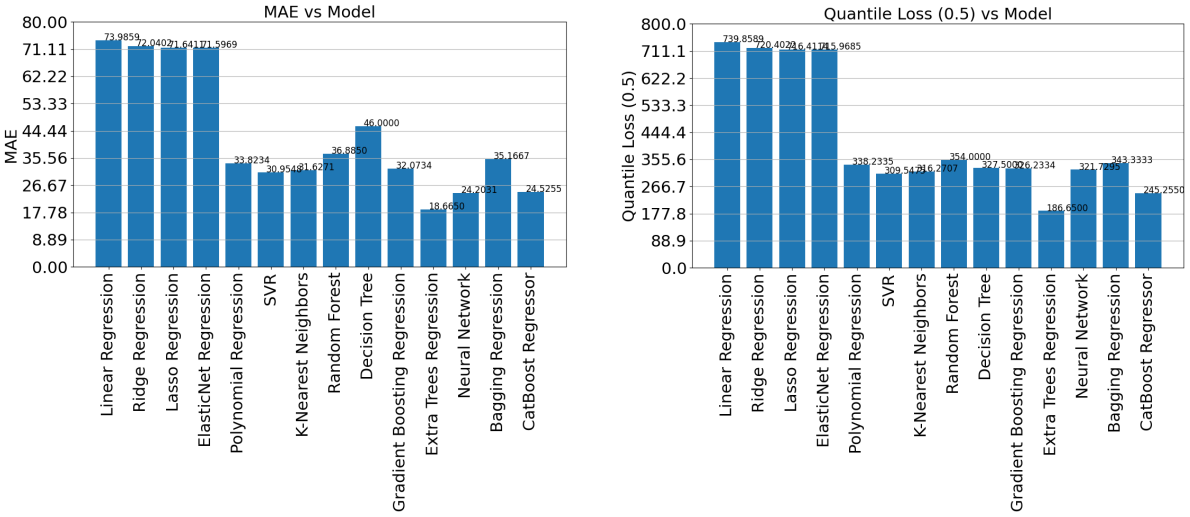
- Coefficient of determination (R^2) [241]
- Mean Absolute Error (MAE) [242]
- Mean Squared Error (MSE) or Root Mean Squared Error (RMSE) [243]
- Quantile loss function
- Expectile loss function



(a) R2 of different ML models

(b) RMSE of different ML models

Figure 4.25: R2 and RMSE of ML models



(a) MAE of different ML models

(b) Quantile values of different ML models

Figure 4.26: MAE and Quantile values of ML models

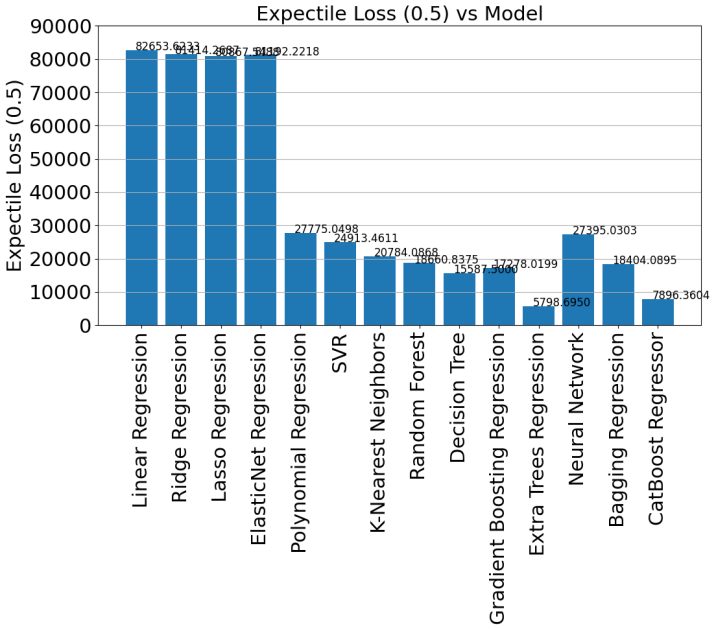


Figure 4.27: Expectile values of different ML models

These ML performance scores are also summarized in the table 4.14 and 4.15.

Table 4.14: Performance of ML models based on R2, RMSE and MAE

Model	RMSE	R2	MAE
Linear Regression	90.914	0.687	73.986
Ridge Regression	90.230	0.692	72.040
Lasso Regression	89.926	0.694	71.641
ElasticNet Regression	90.107	0.693	71.597
Polynomial Regression	52.702	0.895	33.823
SVR	**49.913**	**0.906**	**30.955**
K-Nearest Neighbors	45.590	0.921	31.627
Random Forest	45.170	0.923	36.885
Decision Tree	56.987	0.877	46.000
Gradient Boosting Regression	**41.161**	**0.936**	**32.073**
Extra Trees Regression	**24.080**	**0.978**	**18.665**
Neural Network	**43.292**	**0.929**	**35.167**
Bagging Regression	**41.035**	**0.936**	**24.525**
CatBoost Regression	**28.100**	**0.970**	**35.166**

Table 4.15: Loss Metrics for Different Models

Model	Quantile Loss	Expectile Loss
Linear Regression	739.859	82653.6232
Ridge Regression	720.402	81414.268
Lasso Regression	716.411	80867.548
ElasticNet Regression	715.969	81192.2218
Polynomial Regression	338.234	27775.0497
SVR	309.548	24913.0
K-Nearest Neighbors	316.271	20784.46
Random Forest	354.000	18660.87
Decision Tree	327.500	15587.83
Gradient Boosting Regression	326.233	17278.02
Extra Trees Regression	186.650	5798.69
Neural Network	321.730	27395.03
Bagging Regression	343.333	18404.09
CatBoost Regression	245.255	7896.36

Results are concluded in the conclusion section but in short models such as Extra Trees Regression, Gradient Boosting Regression, Bagging Regression, Neural Network, and CatBoost Regression seem to give the best results for the given data. The way these graphs are red is the model with the highest R2 and the least RMSE, MAE, Quantile and Expectile are the best model. Another observation is finding how much data is needed in order to get good accurate results. In other words, what should be the split percentage for training and testing of ML models in order to achieve high R2 and at the same time have enough data left for testing purposes? The general way of split is 80% training and 20 % testing which also proves to be beneficial in this research. If more data is used for training then less data will be available to make a good estimation for ML model accuracy so more data do not always mean better results.

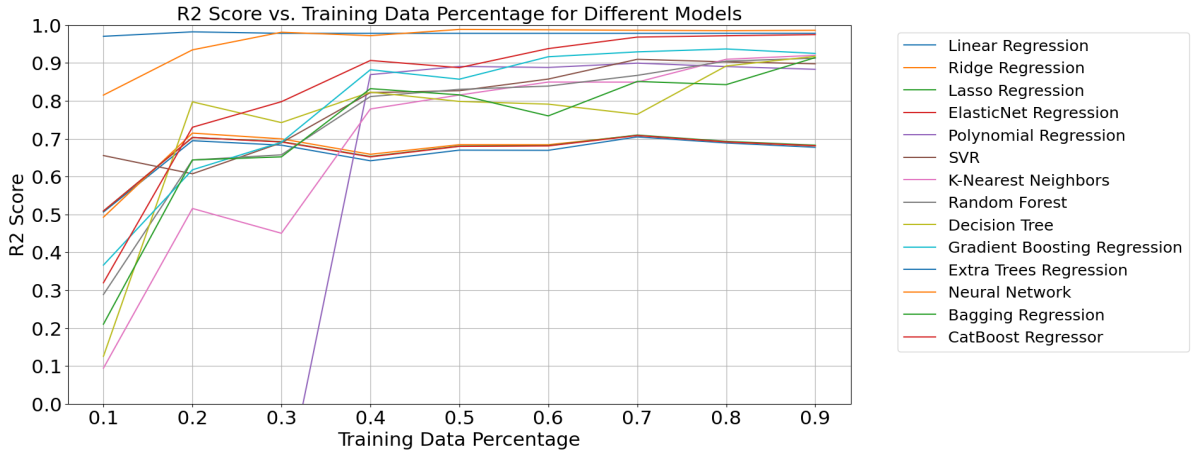
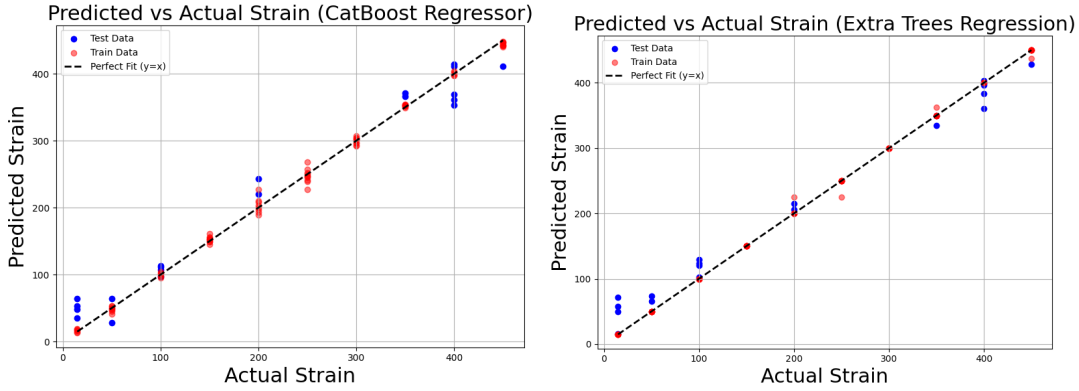


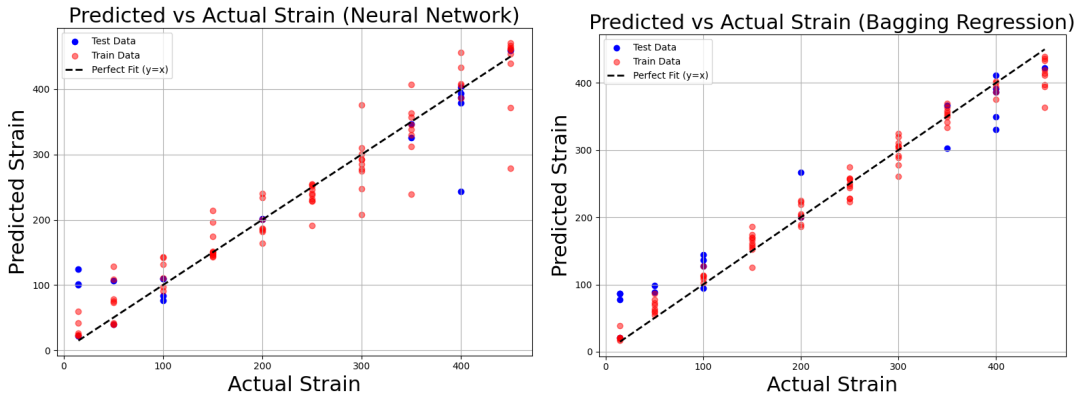
Figure 4.28: R2 [-] vs training data percentage [%]

For the chosen ML models also predicted strain vs actual strain graphs are plotted for few of the chosen ML models.



(a) Predicted vs actual strain using CatBoost Regression **(b)** Predicted vs actual strain using Extra Trees Regression

Figure 4.29: Predicted vs actual strain plot CatBoost and Extra Trees regression



(a) Predicted vs actual strain using Neural Network **(b)** Predicted vs actual strain using Bagging Regression

Figure 4.30: Predicted vs actual strain plot Neural Network and Bagging regression

In the next chapter ”Conclusion and Recommendation” these models are further discussed and analyzed.

4.5. Fatigue life prediction results

In the flowchart as depicted in figure 3.12, strain is estimated already using ML models and loading spectra already plotted as in figure 3.31b. This loading spectra looks like sinusodial waveform since only one strain was used during four-point bending test. In this section, the standard models, as elucidated in Chapter 3 Section 3.5 of the methodology, are employed to predict the fatigue life over a specified number of cycles. This includes predicting fatigue life based on the S-N curve. As previously discussed in Section 4.3.1, strain values were predicted using ML models derived from sensor data. In this section, these predicted strain values are utilized within existing fatigue life prediction models, while the subsequent Section 4.6 is dedicated to the plotting of damage curves using established models. The primary objective of these sections is to demonstrate that once the strain values are known, it becomes possible to predict fatigue life and generate damage curves. This, in turn, offers valuable estimates to engineers regarding the remaining life of the specimen.

In Table 4.5, strain values of 220 $[\frac{\mu m}{m}]$, 160 $[\frac{\mu m}{m}]$, and 114 $[\frac{\mu m}{m}]$, are selected. Given that this research is confined to the realm of constant strain tests, the models detailed in Section 3.5.2 can be directly employed for this purpose.

Table 4.16: Fatigue life as **number of cycles** based on fatigue life models used

Sample number	Strain level (experimental) [µm/m]	Experiment	Cheng	Yingjun Mei et al.	SN-curve	Asphalt institute
1	220	42778	42586	32906	42564	44783
2	160	233831	226863	179870	226749	233084
3	114	1349136	1346096	1037797	1345448	1349199

Table 4.17: %Difference of each model from experimental value

			%Difference			
Sample number	**Strain level (experimental) [µm/m]**	**Experiment**	**Cheng**	**Yingjun Mei et al.**	**SN-curve**	**Asphalt institute**
1	220	42778	-0.45	-23.08	-0.50	4.69
2	160	233831	-2.98	-23.08	-3.03	-0.32
3	114	1349136	-0.23	-23.08	-0.27	0.004

Table 4.5 reveals that the model proposed by Cheng performs exceptionally well. It is evident that the model by Yingjun Mei et al. yields a constant percentage difference, as this model is used only at room temperature and employs a constant divisor. The S-N curve model exhibits slightly inferior performance compared to Cheng's model, while the asphalt institute model demonstrates a 4.68% difference for sample number 1, though it predicts similar values to Cheng's model for sample numbers 2 and 3.

4.6. Damage curves

The methodology is expounded by means of a flowchart, as depicted in Figure 3.12. This methodology has led us to a juncture where we possess estimated strain values, obtained through ML models, and loading spectra, as illustrated in Figure 3.31b, which represents sinusoidal loading in the context of constant strain testing. These strain values have been employed in Section 4.5 to predict fatigue life, and in this section, they are visualized with the aid of models as delineated in Section 3.6, allowing for the plotting of damage curves.

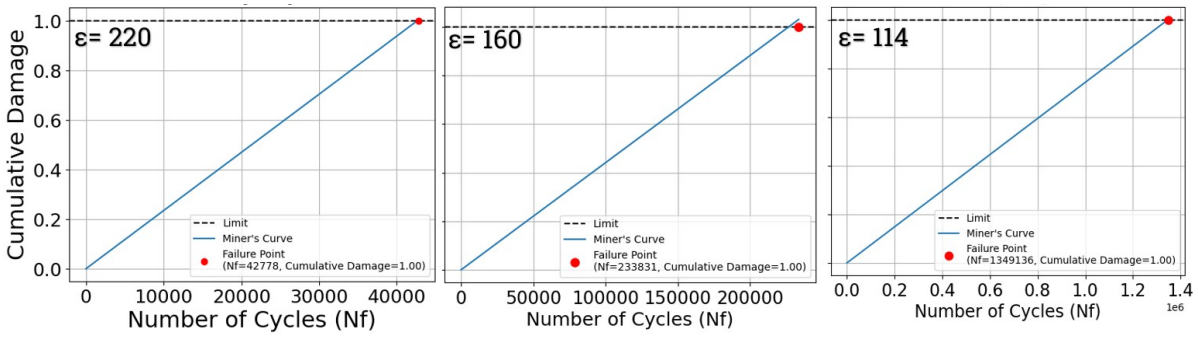


Figure 4.31: Damage vs number of cycles according to Miner's rule

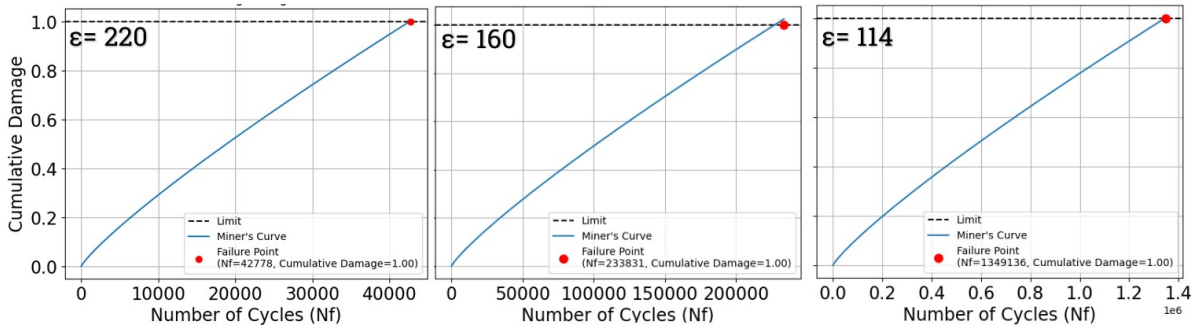


Figure 4.32: Damage vs number of cycles according to Miner's rule with Hopman adjustment

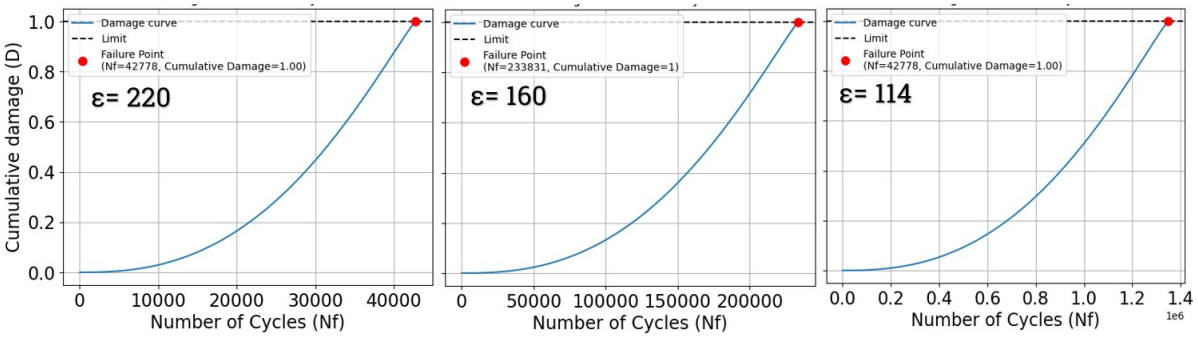


Figure 4.33: Cumulative damage vs number of cycles according to Chahboch and Lasne

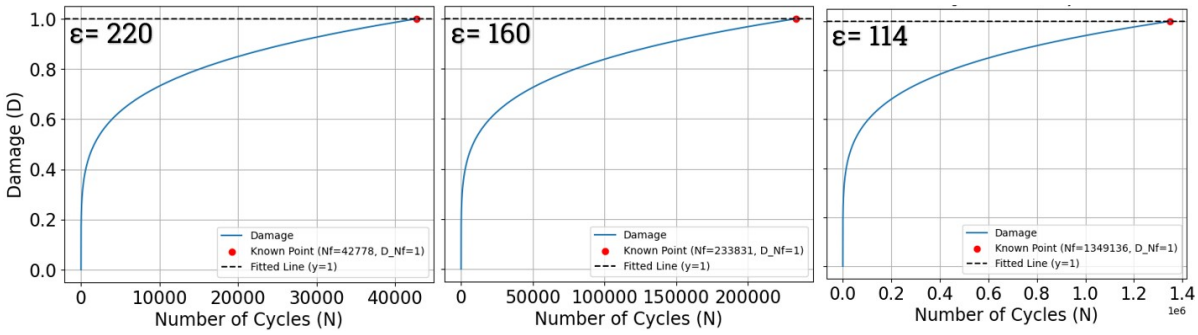


Figure 4.34: Cumulative damage vs number of cycles according to Bodin

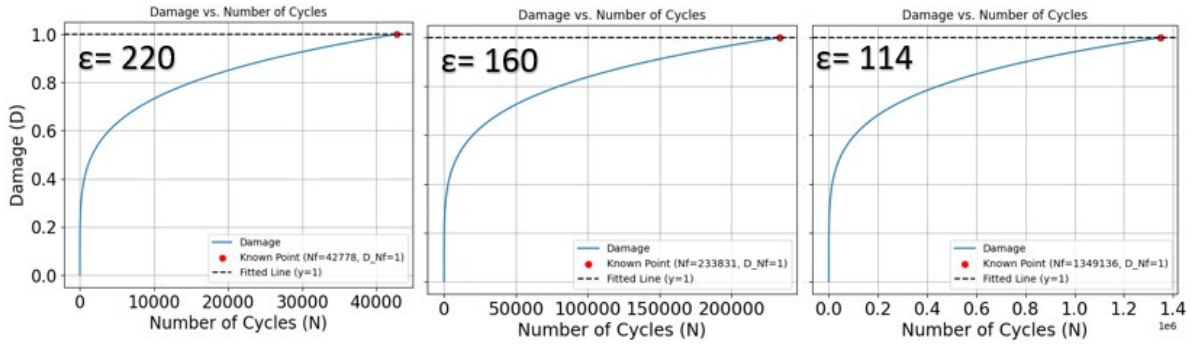


Figure 4.35: Cumulative damage vs number of cycles according to Ma et al.

The damage curves obtained through the utilization of the Ma et al. model and the Bodin model exhibit strikingly similar patterns and outperform other models. However, further research and investigation at the material level are imperative, both for assessing the performance of these models and potentially proposing new models employing a methodology akin to that employed in this study. This additional research is essential to ultimately determine which model performs better, particularly when looking at material level.

4.7. Conclusion

In this chapter, a diverse range of results is presented, all of which were obtained by adhering to the methodology developed in Chapter 3. Prior to conducting the four-point bending (4PB) test on asphalt beams equipped with PZT sensors, a Teflon beam with a PZT sensor was subjected to the same test. The analysis of the PZT sensor data revealed that an increase in strain level or loading frequency during the 4PB test leads to a corresponding increase in voltage amplitude. However, if integral voltage amplitude vs maximum strain is plotted then the results stays same until 5[Hz] as illustrated in figure D.4.

Subsequently, 4PB tests were conducted on asphalt beams, both with and without PZT sensors. The tests without PZT sensors were aimed at constructing the S-N curve and master curve without risking breaking of PZT sensor. Simultaneously, the tests with PZT sensor were conducted to verify whether they exhibit a similar pattern to that observed in the Teflon beam. Upon confirming that the voltage amplitudes generated by PZT sensor in asphalt beams and the Teflon beam follow a comparable pattern, the voltage amplitudes for three different strain levels on asphalt beam, along with their respective frequencies, were employed to estimate strain. These estimates were obtained using pre-trained machine learning (ML) models, with the CatBoost Regression model delivering the most favorable results. These ML models were initially trained on sensor data from the Teflon beam.

Subsequently, fatigue life was predicted using the strain level. Given that constant strain tests were conducted, only one strain level was utilized for fatigue life prediction and for the construction of the damage curve.

5

Conclusion and Recommendation

5.1. Conclusion

In conclusion, this master's thesis embarked on the exploration of non-destructive methods, presenting the innovative idea of employing piezoelectric sensors by attaching them to the substrate in four-point bending for Structural Health Monitoring (SHM). The study involved conducting four-point bending (4PB) tests within the Pavement Engineering laboratory at TU Delft, with sensors attached to Teflon beams. Subsequently, a wealth of data from both PZT and PVDF sensors was collected using an oscilloscope. It was observed that PVDF, due to its petite size, contributed negligibly to the dataset, given the predominance of noise over voltage output. Therefore, only PZT data was used, ignoring PVDF data. This dataset was instrumental in the training and optimization of various machine learning (ML) models to accommodate a broad spectrum of voltage amplitude and its frequency combinations. The 4PB tests employed constant strain testing at different frequencies and strain levels, as the machine applied a constant strain sinusoidal load to the beams, causing the substrate and sensor to bend and generate voltage. The results of these experiments demonstrated a linear correlation between strain and voltage, with higher strain levels or frequency levels yielding increased voltage output. These findings suggest that when incorporated into pavement, these sensors can accurately quantify both the number and type of passing vehicles or alternatively, estimate flexural tensile strain using ML models. The performance in terms of prediction quality of these ML models was rigorously assessed using various metrics and tools such as coefficient of determination (R^2), mean absolute error (MAE), mean squared error (MSE), quantile and expected losses. The analysis leads to the conclusion that, when considering a blend of factors encompassing training duration, predictive accuracy, and the scope for future refinements, the Extra Trees Regression, Neural Network, Catboost Regression, Gradient Boosting Regression, and Bagging Regression emerge as the most efficacious machine learning (ML) models.

It's important to note that these models are likely to improve in performance as the dataset grows. This improvement is due to their built-in adaptability and capability to incorporate insights from previous iterations. Furthermore, this prospect underscores the potential for underperforming ML models to achieve heightened precision and improved quality, providing an opportunity for less successful ML models to refine and enhance their precision and quality, closely tied to the quality and quantity of the training data. In essence, this highlights the prospect for ongoing improvement in ML model performance with the accumulation of higher quality and larger datasets, offering promising prospects for future advancements. However, in conclusion, based on the data gathered, Extra Trees Regression, Bagging Regression, Neural Network, Catboost Regression, and Gradient Boosting Regression are the models that performed better in this study in terms of prediction accuracy, some of them even taking a long time to tune their parameters because once tuned, they take less time for prediction purposes. The hyperparameters selected were standard choices for their respective machine-learning models, guided by the understanding that an excessive number of parameters or extensive options for a single

parameter can significantly prolong the time required for parameter tuning and optimal hyperparameter selection.

The methodology developed in this study and its approach are notably innovative, representing a unique fusion of machine learning and piezoelectric sensors in a previously unexplored manner. While the foundational knowledge of machine learning algorithms is not novel, the application of machine learning in this specific context introduces a groundbreaking concept. Promising results have been achieved through training machine learning models with sensor data from 4PB tests using Teflon beams and predicting strain from Asphalt beams, with Catboost regression emerging as the most effective model. These outcomes highlight the significant potential for future research and the utilization of piezoelectric sensors in in-service pavement. Furthermore, the research has demonstrated that voltage amplitude and its frequency from PZT sensors during Asphalt beam testing exhibit similar patterns to those observed in Teflon beams. Due to limitation of Asphalt beams, more tests are recommended in order to ensure repeatability of observed pattern since now conclusion is made on limited tests on Asphalt beams. Additionally, COMSOL numerical analysis was conducted, further confirming the consistency of sensor data patterns observed in the Teflon beams.

This study is presently in its research stage, and it is evident that further extensive research and testing are imperative. The potential lies in the expansion of this research from its current scope to encompass simulation and real-world applications. Such an extension holds the promise of delivering substantial benefits to various stakeholders responsible for constructing new roads, maintaining existing infrastructure, including bridges, and exploring potential applications in other domains. Ultimately, the broader society stands to gain significantly from the potential of this technology once it receives the necessary approvals and is deployed for practical use.

5.2. Recommendation

In light of this research's pioneering nature and its findings, several key recommendations are offered for future endeavors. Foremost among these is the imperative need for comprehensive material testing to reinforce the validation of Teflon and Asphalt as representative materials since they had similar flexural stiffness in the current study. Testing materials with different flexural stiffness may create different models and may lead to more improvement of the current methodology. Given the study's limitations, stemming from the scarcity of Asphalt beams and potential equipment improvements, a thorough evaluation is essential to confirm their suitability within the proposed methodology. This assessment should extend to addressing equipment adequacy, particularly regarding transducer performance, to ensure research reliability.

Another crucial aspect is the prioritization of the investigation and refinement of sensor attachment methods, considering their observable influence on research outcomes. This also opens the door for the exploration of new attachment designs suitable for potential use in in-service pavement. Additionally, a recommendation is to study sensor and substrate behavior over the long term while simultaneously monitoring at the material level. This approach would provide more insight into material behavior and its degradation over time, generating a deeper understanding of piezoelectric sensor behavior.

Furthermore, it is suggested that future research should consider applying this methodology for varying strain levels instead of relying solely on homogeneous or constant strain tests. While this study primarily focuses on predictive and estimation models, it is recommended for future research to delve into the assessment of material degradation over time, following the methodology proposed here and possibly developing intricate fatigue life models based on sensor data. Comparing the results of using these sensors on pavements with the outcomes of this study or laboratory experiments offers another valuable opportunity for further investigation.

In summary, this research has laid the foundation for innovative applications of piezoelectric sensors and machine learning in the context of structural health monitoring. Nevertheless, a multitude of unexplored opportunities and challenges await in this field. The future holds the promise of further innovations and refinements, providing an exciting pathway for both research and practical implementation.

References

[1] 123RF. "cracked asphalt old as background". In: (). URL: https://www.123rf.com/photo_102104297_cracked-asphalt-old-as-background.html?is_plus=1%5Candorigin=1.

[2] R. Lundstrom, H. Di Benedetto, and U. Isacsson. "Influence of Asphalt Mixture Stiffness on Fatigue Failure". In: Journal of Materials in Civil Engineering 16.6 (2004), pp. 516–525.

[3] G.G. Al-Khateeb and K.A. Ghuzlan. "The Combined Effect of Loading Frequency, Temperature, and Stress Level on the Fatigue Life of Asphalt Paving Mixtures Using the IDT Test Configuration". In: International Journal of Fatigue 59 (2014), pp. 254–261.

[4] U.A. Mannan, M.R. Islam, and R.A. Tarefder. "Effects of recycled asphalt pavements on the fatigue life of asphalt under different strain levels and loading frequencies". In: International Journal of Fatigue 78 (2015), pp. 72–80.

[5] H. Cheng. "Determination of the Stiffness Moduli and Fatigue Endurance Limits of Asphalt Pavements for Perpetual Pavement Design". PhD thesis. Hong Kong: The Hong Kong Polytechnic University, 2022.

[6] F. P. Pramesti, A. A. A. Molenaar, and M. F. C. Van de Ven. The Prediction of Fatigue Life Based on Four Point Bending Test. Manuscript. (author). 2013. DOI: 10.1016/j.proeng.2013.03.078. URL: http://resolver.tudelft.nl/uuid:618c03c1-65ed-4bb1-bdf1-3b0140268bda.

[7] F.P. Pramesti, A.A.A. Molenaar, and M.F.C. van de Ven. "Fatigue Cracking of Gravel Asphalt Concrete: Cumulative Damage Determination". In: 7th RILEM International Conference on Cracking in Pavements. Ed. by A. Scarpas et al. RILEM. 2012, pp. 739–749.

[8] Huailei Cheng et al. "Fatigue test setups and analysis methods for asphalt mixture: A state-of-the-art review". In: Journal of Road Engineering 2.4 (2022), pp. 279–308. ISSN: 20970498. DOI: 10.1016/j.jreng.2022.11.002.

[9] Shogo Morichika et al. "Fatigue Crack Detection Using a Piezoelectric Ceramic Sensor". In: Welding in the World 64 (2020), pp. 141–149. DOI: 10.1007/s40194-019-00807-z. URL: https://doi.org/10.1007/s40194-019-00807-z.

[10] Provincial budget for road maintenance and construction 1.2 billion euro. 2011. URL: https://www.cbs.nl/en-gb/news/2011/07/provincial-budget-for-road-maintenance-and-construction-1-2-billion-euro (visited on 03/20/2023).

[11] Ann M. Johnson and P.E. Best Practices Handbook on Asphalt Pavement Maintenance. Minnesota T2 LTAP Program, Center for Transportation Studies, University of Minnesota: Publisher, 2000.

[12] Wegen. URL: https://www.rijksoverheid.nl/onderwerpen/wegen/vraag-en-antwoord/soorten-wegen-wegbeheerders (visited on 03/23/2023).

[13] Yuval Hernik. STRAIN GAGES AND STRUCTURAL HEALTH MONITORING. Micro managements A VPG brand. July 2021. URL: https://strainblog.micro-measurements.com/content/strain-gages-and-structural-health-monitoring.

[14] H. Hasni et al. "Damage Progression Identification in Asphalt Concrete Pavements: A Smart Self-Powered Sensing Approach". In: Advances in Materials and Pavement Performance Prediction. Ed. by E. Masad et al. Taylor and Francis Group, 2018. ISBN: 978-1-138-31309-5.

[15] Karim Chatti et al. "Damage Detection in Pavement Structures Using Self-powered Sensors". In: 8th RILEM International Conference on Mechanisms of Cracking and Debonding in Pavements. RILEM Bookseries. 2016. Chap. Chapter 93, pp. 665–671. ISBN: 978-94-024-0866-9 978-94-024-0867-6. DOI: 10.1007/978-94-024-0867-6_93.

[16] Y-K. An, M. K. Kim, and H. Sohn. “Piezoelectric Transducers for Assessing and Monitoring Civil Infrastructures”. In: Book Title. Woodhead Publishing Limited, 2014. DOI: 10.1533/9780857099136.86. URL: https://www.example.com/link-to-the-book-chapter.

[17] Demi Ai et al. “Mechanical Impedance Based Embedded Piezoelectric Transducer for Reinforced Concrete Structural Impact Damage Detection: A Comparative Study”. In: Journal Name Volume Number (2018), Page numbers. DOI: 10.0000/your-doi-number. URL: https://www.example.com/link-to-the-article.

[18] Amir H. Alavi et al. “An intelligent structural damage detection approach based on self-powered wireless sensor data”. In: Automation in Construction 62 (2016), pp. 24–44. ISSN: 09265805. DOI: 10.1016/j.autcon.2015.10.001.

[19] J. Braunfelds et al. “Road Pavement Structural Health Monitoring by Embedded Fiber-Bragg-Grating-Based Optical Sensors”. In: Sensors (Basel) 22.12 (2022). Braunfelds, Janis Senkans, Ugis Skels, Peteris Janeliukstis, Rims Porins, Jurgis Spolitis, Sandis Bobrovs, Vjaceslavs. DOI: 10.3390/s22124581. URL: https://www.ncbi.nlm.nih.gov/pubmed/35746362.

[20] Ze Jiao Dong et al. “Asphalt Pavement Structural Health Monitoring Utilizing FBG Sensors”. In: Advanced Engineering Forum 5 (2012), pp. 339–344. ISSN: 2234-991X. DOI: 10.4028/www.scientific.net/AEF.5.339.

[21] Gyuhae Park, Harley H. Cudney, and Daniel J. Inman. “An Integrated Health Monitoring Technique Using Structural Impedance Sensors”. In: Journal of Intelligent Material Systems and Structures 11.6 (2016), pp. 448–455. ISSN: 1045-389X 1530-8138. DOI: 10.1106/qxmv-r3gc-vxxg-w3aq.

[22] S. Park et al. “Multiple Crack Detection of Concrete Structures Using Impedance-based Structural Health Monitoring Techniques”. In: Experimental Mechanics 46.5 (2006), pp. 609–618. ISSN: 0014-4851 1741-2765. DOI: 10.1007/s11340-006-8734-0.

[23] F. G. Baptista and J. V. Filho. “A New Impedance Measurement System for PZT-Based Structural Health Monitoring”. In: IEEE Transactions on Instrumentation and Measurement 58.10 (2009), pp. 3602–3608. ISSN: 0018-9456 1557-9662. DOI: 10.1109/tim.2009.2018693.

[24] Wei Yan and W. Q. Chen. “Structural Health Monitoring Using High-Frequency Electromechanical Impedance Signatures”. In: Advances in Civil Engineering 2010 (2010), pp. 1–11. ISSN: 1687-8086 1687-8094. DOI: 10.1155/2010/429148.

[25] F. G. Baptista et al. “An experimental study on the effect of temperature on piezoelectric sensors for impedance-based structural health monitoring”. In: Sensors (Basel) 14.1 (2014). Baptista, Fabricio G Budoya, Danilo E de Almeida, Vinicius A D Ulson, Jose Alfredo C. DOI: 10.3390/s140101208. URL: https://www.ncbi.nlm.nih.gov/pubmed/24434878.

[26] Ramesh Gomasa et al. “A Review on Health Monitoring of Concrete Structures Using Embedded Piezoelectric Sensor”. In: Construction and Building Materials 305 (2023), p. 133179. ISSN: 0950-0618. DOI: 10.1016/j.conbuildmat.2023.133179. URL: https://doi.org/10.1016/j.conbuildmat.2023.133179.

[27] Pavla Martinkova et al. “Main streams in the Construction of Biosensors and Their Applications”. In: International journal of electrochemical science 12 (Aug. 2017), pp. 7386–7403. DOI: 10.20964/2017.08.02.

[28] T. Jan et al. Fundamentals of Piezoelectric Sensorics: Mechanical, Dielectric, and Thermodynamical Properties of Piezoelectric Materials. New York: Springer, 2010.

[29] W. Staszewski, C. Boller, and G. Tomlinson. Health Monitoring of Aerospace Structures: Smart Sensor Technologies and Signal Processing. England: John Wiley and Sons, Ltd., 2004.

[30] PI Ceramic. Piezoelectric Ceramic Products catalogue. Tech. rep. Lexington, Ky., Lederhose, Germany.

[31] University of Cambridge. PZT. Accessed on 10 octuber 2023. DoITPoMS - Dissemination of IT for the Promotion of Materials Science. Year not specified. URL: https://www.doitpoms.ac.uk/tlplib/piezoelectrics/pzt.php.

[32] Wikipedia Contributors. Lead zirconate titanate. Sept. 2023. URL: https://en.wikipedia.org/wiki/Lead_zirconate_titanate.

[33] W. Zhang, H. Ye, and R. Xiong. "Metal-organic coordination compounds for potential ferroelectrics". In: Coordination Chemistry Reviews 253.23-24 (2009), pp. 2980–2997. DOI: 10.1016/j.ccr.2009.06.003.

[34] University of Cambridge. Polarisation. Accessed on 10 octuber 2023. DoITPoMS - Dissemination of IT for the Promotion of Materials Science. Year not specified. URL: https://www.doitpoms.ac.uk/tlplib/piezoelectrics/polarisation.php.

[35] Piezoelectricity. IEEE, 1987.

[36] J. Sirohi and I. Chopra. "Fundamental understanding of piezoelectric strain sensors". In: Journal of Intelligent Material Systems and Structures 11.4 (2000), pp. 246–257. DOI: 10.1177/104538900911400402.

[37] Jayant Sirohi and Inderjit Chopra. "Fundamental Understanding of Piezoelectric Strain Sensors". In: Journal of Intelligent Material Systems and Structures 11.4 (2000). Downloaded from http://jim.sagepub.com at UNIV OF MARYLAND on August 3, 2009, pp. 246–257. ISSN: 1530-8138. DOI: 10.1106/8BFB-GC8P-XQ47-YCQ0. URL: http://jim.sagepub.com.

[38] Hua-Ping Wang et al. "Review on structural damage rehabilitation and performance assessment of asphalt pavements". In: Reviews on Advanced Materials Science 60.1 (2021), pp. 438–449. ISSN: 1605-8127. DOI: 10.1515/rams-2021-0030.

[39] M. Motamedi, G. Shafabakhsh, and M. Azadi. "Evaluation of fatigue and rutting properties of asphalt binder and mastic modified by synthesized polyurethane". In: Journal of Traffic and Transportation Engineering (English Edition) 8.6 (2021), pp. 1036–1048.

[40] H. Wang. "Strain transfer of optical fiber under damage conditions and its application in multi-layered pavements". PhD thesis. Dalian University of Technology, 2015.

[41] A. Garcia, J. Jelfs, and C. J. Austin. "Internal asphalt mixture rejuvenation using capsules". In: Construction and Building Materials 101.1 (2015), pp. 309–316.

[42] F.N. Hveem. "Pavement deflections and fatigue failures". In: Highway Research Board Bulletin 114 (1955), pp. 43–87.

[43] Y. Huang. Pavement Analysis and Design: United States Edition. Upper Saddle River: Prentice-Hall, 1993.

[44] Y.R. Kim. Modeling of Asphalt Concrete. Reston: ASCE Press, 2008.

[45] L. Sun. Structural Behavior of Asphalt Pavements. Kidlington: Butterworth-Heinemann, 2016.

[46] Y. Wang, Y. Wen, H. Cheng, et al. "Endurance strain limits of long-life flexible pavements". In: China Journal of Highway and Transport 33.10 (2020), p. 102.

[47] H.P. Bell, I.L. Howard, R.B. Freeman, et al. "Evaluation of remaining fatigue life model for hot-mix asphalt airfield pavements". In: International Journal of Pavement Engineering 13.4 (2012), pp. 281–296.

[48] S. Wei, Y. Wang, H. Cheng, et al. "Stress distributions in the textures of prefabricated pavement surface created with the assistance of 3D printing technology". In: International Journal of Pavement Engineering (2021). DOI: 10.1080/10298436.2021.2005058.

[49] L. Sun, J. Yuan, T. Ding, et al. "Elaboration of a damage monitor and assessment approach for in-situ asphalt pavement layer using portable seismic property analyzer (PSPA)". In: NDTInternational 131 (2022), p. 102692.

[50] What Causes Fatigued Cracking in Asphalt? 2016. URL: https://www.aciindiana.com/blog/what-causes-fatigued-cracking-in-asphalt/#:~:text=A%5C%20scattered%5C%20series%5C%20of%5C%20interconnected,repetitive%5C%20loading%5C%20and%5C%20heavy%5C%20traffic (visited on 05/27/2023).

[51] Y.H. Huang. Pavement Analysis and Design. second. Hoboken: Prentice-Hall, 2003.

[52] Asphalt Damage Types: Block, Fatigue and Linear Cracking. URL: https://gopaveutah.com/asphalt-damage-types-block-fatigue-and-linear-cracking/ (visited on 05/27/2023).

[53] Fatigue Cracking. URL: https://pavementinteractive.org/reference-desk/pavement-management/pavement-distresses/fatigue-cracking (visited on 05/26/2023).

[54] Nithin Sudarsanan and Youngsoo Richard Kim. "A critical review of the fatigue life prediction of asphalt mixtures and pavements". In: Journal of Traffic and Transportation Engineering (English Edition) 9.5 (2022), pp. 808–835. ISSN: 20957564. DOI: 10.1016/j.jtte.2022.05.003.

[55] Top-Down Cracking of Hot-Mix Asphalt Layers: Models for Initiation and Propagation (Web-Only Document No. 162). Washington DC, 2010.

[56] J.A.F. Harvey and D. Cebon. "Fracture tests on bitumen films". In: Journal of Materials in Civil Engineering 17 (2005), pp. 99–106.

[57] J.A. Deacon et al. "Temperature considerations in asphalt-aggregate mixture analysis and design". In: Transportation Research Record 1454 (1994), pp. 97–112.

[58] HMA Pavement. URL: https://pavementinteractive.org/reference-desk/pavement-types-and-history/pavement-types/hma-pavement/ (visited on 05/28/2023).

[59] Y. Wang, A.S. Wong, Y. Wen, et al. "Characterization of the distress modes and in situ material properties of highway asphalt pavement with long service life". In: Journal of Performance of Constructed Facilities 30.4 (2016), p. 4015095.

[60] Islam. "THERMAL FATIGUE DAMAGE OF ASPHALT PAVEMENT". In: 2015.

[61] CCRB Science and Technology Co., Ltd. Specifications for Design of Highway Asphalt Pavement. JTG D50-2017. Beijing: China Communications Press, 2017.

[62] National Cooperative Highway Research Program, ed. Guide for Mechanistic-Empirical Design of New and Rehabilitated Pavement Structures. NCHRP Report 1-37A. Washington DC: TRB, 2004.

[63] C.L. Monismith and J.A. Deacon. "Fatigue of asphalt paving mixtures". In: Transportation Engineering Journal of ASCE 95.2 (1969), pp. 317–346.

[64] P.S. Pell. "Fatigue characteristics of bitumen and bituminous mixes". In: International Conference on the Structural Design of Asphalt Pavements. Ann Arbor, 1962.

[65] H. Cheng, J. Liu, L. Sun, et al. "Fatigue behaviours of asphalt mixture at different temperatures in four-point bending and indirect tensile fatigue tests". In: Construction and Building Materials 273 (2021), p. 121675.

[66] Asphalt Institute. Thickness Design: Full Depth Asphalt Pavement Structures for Highways and Streets. eighth. College Park: The Asphalt Institute, 1991.

[67] C.L. Monismith, J.A. Epps, and F.N. Finn. "Improved asphalt mix design (with discussion)". In: Journal of Association Asphalt Paving Technol 54 (1985), pp. 340–406.

[68] Shell International Petroleum Company. Shell Pavement Design Manual: Asphalt Pavements and Overlays for Road Traffic. Tech. rep. London, 1978.

[69] Z. Si, D.N. Little, and R.L. Lytton. "Characterization of microdamage and healing of asphalt concrete mixtures". In: Journal of Materials in Civil Engineering 14 (2002), pp. 461–470.

[70] Enwuso A. Igwe. "Developing a Comparative Model for Predicting Fatigue Life of Candle Wax Modified Hot Mix Asphalt Concrete Mixtures: Heavy Traffic Case". In: (2014).

[71] H. Wen. "Use of fracture work density obtained from indirect tensile testing for the mix design and development of a fatigue model". In: International Journal of Pavement Engineering 14.6 (2013), pp. 561–568.

[72] A. Bahadori, A. Mansourkhaki, and M. Ameri. "A phenomenological fatigue performance model of asphalt mixtures based on fracture energy density". In: Journal of Testing and Evaluation 43.1 (2014), p. 20130057.

[73] M. Castro and J.A. Sánchez. “Estimation of asphalt concrete fatigue curves – a damage theory approach”. In: Construction and Building Materials 22.6 (2008), pp. 1232–1238.

[74] Z. Ma, L. Liu, Y. Yuan, et al. “Estimation of total fatigue life for in-service asphalt mixture based on accelerated pavement testing and four-point bending beam fatigue tests”. In: Canadian Journal of Civil Engineering 46.7 (2019), pp. 557–566.

[75] M.A. Miner. “Cumulative damage in fatigue”. In: Journal of Applied Mechanics 12.3 (1945), pp. 159–164.

[76] P. Hopman, P. Kunst, and A. Pronk. “A renewed interpretation method for fatigue measurements-verification of Miner’s rule”. In: The 4th Eurobitume Symposium. Madrid, 1989.

[77] María Castro and José A. Sánchez. “Estimation of asphalt concrete fatigue curves – A damage theory approach”. In: Construction and Building Materials 22.6 (2008), pp. 1232–1238. ISSN: 09500618. DOI: 10.1016/j.conbuildmat.2007.01.012.

[78] Yingjun MEI et al. “Fatigue Life Prediction of Asphalt Pavement Based on Cumulative Damage”. In: Journal of [Journal Name] [Volume Number].[Issue Number] ([Year]), [Page Range].

[79] H. Di Benedetto, C. de La Roche, H. Baaj, et al. “Fatigue of bituminous mixtures”. In: Materials and Structures 37.3 (2004), pp. 202–216.

[80] J. Chaboche and P. Lesne. “A non-linear continuous fatigue damage model”. In: Fatigue and Fracture of Engineering Materials and Structures 11.1 (1988), pp. 1–17.

[81] D. Bodin, G. Pijaudier-Cabot, C. de La Roche, et al. “A continuum damage approach of asphalt concrete fatigue tests”. In: Engineering Mechanics Conference. New York, 2002.

[82] F. Bonnaure, A. Gravois, and J. Udron. “A new method for predicting the fatigue life of bituminous mixes”. In: Association of Asphalt Paving Technologists Proceedings 49 (1980), pp. 499–529.

[83] T. O. Medani and A. A. A. Molenaar. “Fatigue Characteristics of Asphalt Mixes”. In: Heron 45.3 (2000), pp. 155–165.

[84] Terhi K. Pellinen et al. “Fatigue-Transfer Functions: How Do They Compare?” In: Transportation Research Record: Journal of the Transportation Research Board 1896.1 (2004), pp. 77–87. ISSN: 0361-1981 2169-4052. DOI: 10.3141/1896-08.

[85] Asphalt Institute. Research and Development of the Asphalt Institute’s Thickness Design Manual (MS-1). Research Report 82-2. Lexington, Ky., 1982.

[86] B.-W. Tsai, J. T. Harvey, and C. L. Monismith. “WesTrack Fatigue Performance Prediction Using Miner’s Law”. In: Transportation Research Record: Journal of the Transportation Research Board 1809 (2002), pp. 137–147.

[87] H. Di Benedetto, C. de La Roche, H. Baaj, et al. “Fatigue of bituminous mixtures: different approaches and RILEM group contribution”. In: Sixth International RILEM Symposium on Performance Testing and Evaluation of Bituminous Materials. Zurich, 2003.

[88] Y. Hu, G. Zang, L. Sun, et al. “Determination of optimal identity points for backcalculating the structural layer moduli of asphalt pavement”. In: Chinese Science Bulletin 65.30 (2020), pp. 3287–3297.

[89] W. Wang, H. Cheng, L. Sun, et al. “Multi-performance evaluation of recycled warm-mix asphalt mixtures with high reclaimed asphalt pavement contents”. In: Journal of Cleaner Production 2022 (2022), p. 134209.

[90] X. Zhu et al. “Rutting and fatigue performance evaluation of warm mix asphalt mastic containing high percentage of artificial RAP binder”. In: Construction and Building Materials 240 (2020), p. 117860.

[91] Krishna Prapoorna Biligiri and Safwat H. Said. “Prediction of the Remaining Fatigue Life of Flexible Pavements Using Laboratory and Field Correlations”. In: Journal of Materials in Civil Engineering 27.7 (2015). ISSN: 0899-1561 1943-5533. DOI: 10.1061/(asce)mt.1943-5533.0001161.

[92] L. Li et al. “A Prediction Model on Viscoelastic Fatigue Damage of Asphalt Mixture”. In: Materials (Basel) 13.17 (2020). ISSN: 1996-1944 (Print) 1996-1944 (Electronic) 1996-1944 (Linking). DOI: 10.3390/ma13173782. URL: https://www.ncbi.nlm.nih.gov/pubmed/32867202.

[93] A. C. Pronk, M. Gajewski, and W. Bańkowski. “Processing of four point bending test results for visco-elasticity and fatigue models”. In: International Journal of Pavement Engineering 20.10 (2017), pp. 1226–1230. ISSN: 1029-8436 1477-268X. DOI: 10.1080/10298436.2017.1398549.

[94] A. Mateos, R. Wu, E. Denneman, et al. “Sine versus haversine displacement waveform comparison for hot mix asphalt four-point bending fatigue testing”. In: Transportation Research Record 2672 (2018), pp. 372–382.

[95] X. Xiao, J. Li, D. Cai, et al. “Characterizing thermal fatigue behaviors of asphalt concrete waterproofing layer in high-speed railway using customized overlay test”. In: International Journal of Fatigue 165 (2022), p. 107176.

[96] M. Witczak, M. Mamlouk, M. Souliman, et al. Laboratory Validation of an Endurance Limit for Asphalt Pavements. Tech. rep. NCHRP Report 762. Washington DC: FHWA, 2013.

[97] J. Deacon, A. Tayebali, J. Coplantz, et al. Fatigue Response of Asphalt-Aggregate Mixes, Part III—Mix Design and Analysis. Tech. rep. 404. Washington DC: SHRP, 1994.

[98] B.W. Tsai, J.T. Harvey, and C.L. Monismith. “High temperature fatigue and fatigue damage process of aggregate-asphalt mixes”. In: Journal of Association of Asphalt Paving Technologists 71.1 (2002), pp. 45–385.

[99] M. Witczak, M. Mamlouk, M. Kaloush, et al. “Validation of initial and failure stiffness definitions in flexure fatigue test for hot mix asphalt”. In: Journal of Testing and Evaluation 35.1 (2007), pp. 95–102.

[100] B. Prowell, E. Brown, R. Anderson, et al. Validating the Fatigue Endurance Limit for Hot Mix Asphalt. Washington DC: Transportation Research Board, 2010.

[101] American Association of State Highway and Transportation Officials (AASHTO). Standard Method of Test for Determining the Fatigue Life of Compacted Hot-Mix Asphalt (HMA) Subjected to Repeated Flexural Bending. AASHTO T321. Washington DC: AASHTO, 2014.

[102] American Association of State Highway and Transportation Officials (AASHTO). Estimating Damage Tolerance of Asphalt Binders Using the Linear Amplitude Sweep. AASHTO TP 101. Washington DC: AASHTO, 2014.

[103] B.W. Tsai. “High-Temperature Fatigue and Fatigue Damage Process of AggregateAsphalt Mixes”. PhD thesis. University of California, Berkeley, 2001.

[104] H.J. Lee, J.S. Daniel, and Y.R. Kim. “Continuum damage mechanics-based fatigue model of asphalt concrete”. In: Journal of Materials in Civil Engineering 12.2 (2000), pp. 105–112.

[105] Y. Richard Kim, Dallas Little, and Robert Lytton. “Fatigue and Healing Characterization of Asphalt Mixtures”. In: Journal of Materials in Civil Engineering 15.1 (2003), pp. 75–83.

[106] S. Shen and S.H. Carpenter. “Application of the dissipated energy concept in fatigue endurance limit testing”. In: Transportation Research Record 1929 (2005), pp. 165–173.

[107] M.E. Kutay, N. Gibson, and J. Youtcheff. “Conventional and viscoelastic continuum damage (VECD)-based fatigue analysis of polymer modified asphalt pavements (with discussion)”. In: Journal of the Association of Asphalt Paving Technologists 77 (2008), pp. 395–434.

[108] S. Shen, H.M. Chiu, and H. Huang. “Characterization of fatigue and healing in asphalt binders”. In: Journal of Materials in Civil Engineering 22.9 (2010), pp. 846–852.

[109] H. Cheng, L. Sun, Y. Wang, et al. “Effects of actual loading waveforms on the fatigue behaviours of asphalt mixtures”. In: International Journal of Fatigue 151 (2021), p. 106386.

[110] F. Moreno-Navarro and M. Rubio-Gámez. “A review of fatigue damage in bituminous mixtures: understanding the phenomenon from a new perspective”. In: Construction and Building Materials 113 (2016), pp. 927–938.

[111] T.M. Ahmed, H. Al-Khalid, and T.Y. Ahmed. "Review of techniques, approaches and criteria of hot-mix asphalt fatigue". In: Journal of Materials in Civil Engineering 31.12 (2019), p. 3119004.

[112] Y. Zhang, L. Sun, and H. Cheng. "Laboratory performance evaluation of hotmix asphalt mixtures with different design parameters". In: Applied Sciences 10.9 (2020), p. 3038.

[113] Y. Zhang, L. Sun, and H. Cheng. "Effects of nominal maximum aggregate size and compaction effort on the mechanical properties of hot-mix asphalt (HMA)". In: Construction and Building Materials 324 (2022), p. 126715.

[114] Tu Delft. Facilities. 2002. URL: https://www.tudelft.nl/citg/over-faculteit/afdelingen/engineering-structures/sections-labs/pavement-engineering/laboratory/facilities (visited on 06/01/2023).

[115] AASHTO. Standard Method of Test for Determining the Damage Characteristic Curve of Asphalt Mixtures from Direct Tension Cyclic Fatigue Tests. Tech. rep. AASHTO TP107. Washington DC: American Association of State and Highway Transportation Officials, 2018.

[116] AASHTO. Standard Method of Test for Determining Dynamic Modulus of Hot-Mix Asphalt Concrete Mixtures. Tech. rep. AASHTO T342. Washington DC: American Association of State and Highway Transportation Officials, 2019.

[117] AASHTO. Standard Method of Test for Determining the Fatigue Life of Compacted Asphalt Mixtures Subjected to Repeated Flexural Bending. Tech. rep. AASHTO T321. Washington DC: American Association of State and Highway Transportation Officials, 2017.

[118] ASTM. Standard Test Method for Determining Fatigue Failure of Compacted Asphalt Concrete Subjected to Repeated Flexural Bending. ASTM D7460. Washington DC: ASTM, 2010.

[119] European Committee for Standardization (CEN). Bituminous Mixtures–Test Methods–Part 24: Resistance to Fatigue. EN 12697-24. Brussels: CEN, 2018.

[120] AFNOR. Test Relating to Pavements. Measurement of Rheological Properties on Bituminous Mixes. Part 2: Determination of the Dynamic Bending Modulus. Tech. rep. NF P98260-2. Cedex: AFNOR, 1992.

[121] BSI. Bituminous Mixtures, Test Methods, Resistance to Fatigue. Tech. rep. EN 12697-24. London: British Standards Institution, 2018.

[122] BSI. Bituminous Mixtures, Test Methods, Stiffness. Tech. rep. EN 12697-26. London: British Standards Institution, 2018.

[123] AASHTO. Standard Method of Test for Determining the Creep Compliance and Strength of Hot-Mix Asphalt (HMA) Using the Indirect Tensile Test Device. Tech. rep. AASHTO T312. Washington DC: American Association of State and Highway Transportation Officials (AASHTO), 2016.

[124] American Association of State Highway and Transportation Officials (AASHTO). Determining the Damage Characteristic Curve and Failure Criterion Using the Asphalt Mixture Performance Tester (AMPT) Cyclic Fatigue Test. AASHTO TP 107. Washington DC: AASHTO, 2018.

[125] H. Cheng, Y. Wang, L. Liu, et al. "Back-calculation of the moduli of asphalt pavement layer using accelerated pavement testing data". In: Accelerated Pavement Testing to Transport Infrastructure Innovation. Cham: Springer, 2020, pp. 379–388.

[126] H. Cheng, L. Liu, and L. Sun. "Bridging the gap between laboratory and field moduli of asphalt layer for pavement design and assessment: a comprehensive loading frequency-based approach". In: Frontiers of Structural and Civil Engineering 16.3 (2022), pp. 267–280.

[127] H. Cheng et al. "Relating field moduli of asphalt mixture layer under vehicular loading and its dynamic moduli under laboratory loading". In: Transportation Research Record 2676 (2022), pp. 567–579.

[128] C. Monismith, J. Epps, D. Kasianchuk, et al. Asphalt Mixture Behavior in Repeated Flexure. Report TE 70-5. University of California, Berkeley: Institute of Transportation and Traffic Engineering, 1972.

[129] N. Li et al. "Investigation into the size effect on four-point bending fatigue tests". In: Four-Point Bending. Ed. by J.C. Pais and J.T. Harvey. London: CRC Press, 2012, pp. 35–47.

[130] D. Bodin, C. de la Roche, and G. Pijaudier-Cabot. "Size effect regarding fatigue evaluation of asphalt mixtures". In: Road Materials and Pavement Design 7 (2006), pp. 181–201.

[131] B.W. Tsai, M. Bejarano, J. Harvey, et al. "Prediction and calibration of pavement fatigue performance using two-stage Weibull approach". In: Journal of the Association of Asphalt Paving Technologists 74 (2005), pp. 697–732.

[132] T.W. Kennedy, G.A. Huber, E.T. Harrigan, et al. Superior Performing Asphalt Pavements (Superpave): the Product of the SHRP Asphalt Research Program. Tech. rep. SHRPA-410. Washington DC: SHRP, 1994.

[133] F.M. Nejad, E. Aflaki, and M.A. Mohammadi. "Fatigue behavior of SMA and HMA mixtures". In: Construction and Building Materials 24.7 (2010), pp. 1158–1165.

[134] K. Raithby and A. Sterling. "Laboratory fatigue tests on rolled asphalt and their relation to traffic loading". In: Roads and Road Construction 50.596/597 (1972), pp. 219–223.

[135] P.S. Baburamani. Asphalt Fatigue Life Prediction Models: a Literature Review. Tech. rep. ARR 334. Vermont South: Australian Road Research Board, 1999.

[136] A. Ongel and J. Harvey. Analysis of 30 Years of Pavement Temperatures Using the Enhanced Integrated Climate Model (EICM). Tech. rep. San Bernardino: California Department of Transportation, 2004.

[137] Fatigue Response of Asphalt-Aggregate Mixtures. Tech. rep. Washington DC, 1994.

[138] D. Bodin, J.P. Terrier, and C. Perroteau. "Effect of temperature on fatigue performances of asphalt mixes". In: 11th International Conference on Asphalt Pavements. Nagoya, 2010.

[139] A. Pronk. Determination of Temperature Increase in UPP Tests. Tech. rep. Delft: RILEM, 2000.

[140] Michael Mamlouk et al. "Refining HMA Beam Fatigue Testing Conditions". In: Advances in Civil Engineering 1.1 (2012). DOI: 10.1520/ACEM20120018.

[141] H. Di Benedetto et al. "Fatigue of bituminous mixtures". In: Materials Structures 37 (2004), pp. 202–216.

[142] A. Mateos, R. Wu, E. Denneman, et al. Evaluation of the Effect of Deflection Waveform on Fatigue Test Results for Hot Mix Asphalt. UCPRC-TM 2015-03. Napa: California Department of Transportation, 2017.

[143] H. Cheng, L. Sun, Y. Wang, et al. "Analysis of fatigue behaviors of asphalt mixture under actual loading waveforms using pseudo-strain-based approaches". In: International Journal of Pavement Engineering (2022). https://doi.org/10.1080/10298436.2021.2020269.

[144] Brian S Everitt and Anders Skrondal. The Cambridge Dictionary of Statistics. Fourth. Cambridge University Press, 2002.

[145] S. Krishnan et al. "A Methodology for Learning, Analyzing, and Mitigating Social Influence Bias in Recommender Systems". In: RecSys. 2014.

[146] Definition of Data-Driven. Cambridge Dictionary. URL: https://dictionary.cambridge.org/dictionary/english/data-driven (visited on 11/16/2022).

[147] Stuart Russell and Peter Norvig. Artificial Intelligence: A Modern Approach. 1994.

[148] Siddhartha Ray Barua. "A Strategic Perspective on the Commercialization of Artificial Intelligence: A Socio-Technical Analysis". Massachusetts Institute of Technology, 2019.

[149] GeeksforGeeks. Types of Regression Techniques in ML. URL: https://www.geeksforgeeks.org/types-of-regression-techniques/.

[150] Cornell University. Lecture 1: Supervised Learning. Accessed on September 24, 2023. URL: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote01_MLsetup.html.

[151] GeeksforGeeks. Introduction to Tree – Data Structure and Algorithm Tutorials. Accessed on September 24, 2023. URL: https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/.

[152] Rohit Sharma. 5 Types of Binary Tree Explained [With Illustrations]. Sept. 2022. URL: https://www.upgrad.com/blog/5-types-of-binary-tree/.

[153] Guilherme Henrique dos Santos. A brief introduction to Decision Trees. Oct. 2021. URL: https://medium.com/@dos_santos_98/a-brief-introduction-to-decision-trees-706391a879c2.

[154] Lior Rokach and Oded Maimon. "Chapter 9: Decision Trees". In: DATA MINING AND KNOWLEDGE DISCOVERY HANDBOOK. Tel-Aviv, Israel: Department of Industrial Engineering, Tel-Aviv University, Publication date not available.

[155] Rosaria Silipo. From a Single Decision Tree to a Random Forest. Oct. 2019. URL: https://towardsdatascience.com/from-a-single-decision-tree-to-a-random-forest-b9523be65147.

[156] Moshood Hambali et al. "ADABOOST ENSEMBLE ALGORITHMS FOR BREAST CANCER CLASSIFICATION". In: (Aug. 2019), pp. 1–10.

[157] Omer Sagi and Lior Rokach. "Ensemble learning: A survey". In: WIREs Data Mining and Knowledge Discovery 8.4 (2018), e1249. DOI: https://doi.org/10.1002/widm.1249. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1249. URL: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1249.

[158] 1.10. Decision Trees. scikit-learn Developers (BSD License). 2022. URL: https://scikit-learn.org/stable/modules/tree.html (visited on 10/25/2022).

[159] Tin Kam Ho. "Random Decision Forests". In: Proceedings of 3rd International Conference on Document Analysis and Recognition. 1995. DOI: 10.1109/ICDAR.1995.598994.

[160] Tomonori Masui. All You Need to Know about Gradient Boosting Algorithm − Part 1. Regression. Jan. 2022. URL: https://towardsdatascience.com/all-you-need-to-know-about-gradient-boosting-algorithm-part-1-regression-2520a34a502.

[161] Robert E Schapire. "The Strength of Weak Learnability". In: Machine Learning 5.2 (1990), pp. 197–227.

[162] Michael Kearns and Leslie Valiant. Learning Boolean Formulae or Finite Automata is as Hard as Factoring. Tech. rep. TR 14-88, 1988.

[163] scikit-learn. scikit-learn: GradientBoostingRegressor. Accessed on September 24, 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html.

[164] Runhua Guo, Donglei Fu, and Giuseppe Sollazzo. "An Ensemble Learning Model for Asphalt Pavement Performance Prediction Based on Gradient Boosting Decision Tree". In: International Journal of Pavement Engineering 23.10 (2021), pp. 1–14. ISSN: 1477268X. DOI: 10.1080/10298436.2021.1910825.

[165] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY: ACM, 2016, pp. 785–794.

[166] 1.11.4. Gradient Tree Boosting. scikit-learn Developers (BSD License). 2022. URL: https://scikit-learn.org/stable/modules/ensemble.html (visited on 10/27/2022).

[167] scikit-learn developers. Ensembles: Gradient boosting, random forests, bagging, voting, stacking. scikit-learn. Year of Access. URL: https://scikit-learn.org/stable/modules/ensemble.html.

[168] IBM. What is Bagging? Accessed on September 24, 2023. URL: https://www.ibm.com/topics/bagging#:~:text=In%201996%2C%20Leo%20Breiman%20(PDF,technique%20to%20create%20diverse%20samples..

[169] Ludmila I. Kuncheva. Combining Pattern Classifiers: Methods and Algorithms. New York, NY: John Wiley and Sons, 2004.

[170] Leo Breiman. "Random Forests". In: Machine Learning 45.1 (2001), pp. 5–32.

[171] Cornell University. 18: Bagging. Accessed on September 24, 2023. URL: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote18.html.

[172] scikit-learn. scikit-learn Documentation: 1.11. Ensembles: Gradient boosting, random forests, bagging, voting, stacking. Accessed on September 24, 2023. URL: https://scikit-learn.org/stable/modules/ensemble.html#b1998.

[173] Pierre Geurts, Damien Ernst, and Louis Wehenkel. "Extremely Randomized Trees". In: Machine Learning 63.1 (2006), pp. 3–42.

[174] Jason Brownlee. How to Develop an Extra Trees Ensemble with Python. Last Updated on April 27, 2021. 2020. URL: https://machinelearningmastery.com/extra-trees-ensemble-with-python/.

[175] Rupak (Bob) Roy. Extra Trees Classifier / Regressor. June 2021. URL: https://bobrupakroy.medium.com/extra-trees-classifier-regressor-5b5f6abe8228.

[176] Yingli Lou et al. "Individualized empirical baselines for evaluating the energy performance of existing buildings". In: Science and Technology for the Built Environment 29 (Oct. 2022), pp. 1–15. DOI: 10.1080/23744731.2022.2134680.

[177] Harshit Kapoor. Random Forest vs Extra Trees. 2020. URL: https://www.kaggle.com/code/hkapoor/random-forest-vs-extra-trees.

[178] Jason Brownlee. How to Develop a Random Forest Ensemble in Python. Last Updated on April 27, 2021. 2020. URL: https://machinelearningmastery.com/random-forest-ensemble-in-python/.

[179] Bradley Efron and Trevor Hastie. Computer Age Statistical Inference, Student Edition: Algorithms, Evidence, and Data Science. Vol. 6. Cambridge University Press, 2021.

[180] V. John et al. "Real-time Lane Estimation Using Deep Features and Extra Trees Regression". In: Image and Video Technology. Springer, 2015, pp. 721–733.

[181] IBM. AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the difference? Accessed on September 24, 2023. URL: https://www.ibm.com/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks/.

[182] Deepika Singh. Machine Learning with Neural Networks Using scikit-learn. June 2019. URL: https://www.pluralsight.com/guides/machine-learning-neural-networks-scikit-learn.

[183] scikit-learn. scikit-learn Documentation: 1.17. Neural network models (supervised). Accessed on September 24, 2023. URL: https://scikit-learn.org/stable/modules/neural_networks_supervised.html.

[184] Dasaradh S K. A Gentle Introduction To Math Behind Neural Networks. Oct. 2020. URL: https://towardsdatascience.com/introduction-to-math-behind-neural-networks-e8b60dbbdeba.

[185] Cornell University. 20: Neural Network. Accessed on September 24, 2023. URL: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote20.html.

[186] Ajitesh Kumar. Instance-based vs Model-based Learning: Differences. Dec. 2022. URL: https://vitalflux.com/instance-based-learning-model-based-learning-differences/.

[187] GeeksforGeeks. Instance-based learning. Accessed on September 24, 2023. URL: https://www.geeksforgeeks.org/instance-based-learning/.

[188] Tavish Srivastava. A Complete Guide to K-Nearest Neighbors (Updated 2023). Sept. 2023. URL: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/.

[189] Cornell University. Lecture 2: k-nearest neighbors. Accessed on September 24, 2023. URL: https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote02_kNN.html.

[190] V. N. Vapnik. Statistical Learning Theory. Wiley, 1998.

[191] H. Drucker et al. "Support Vector Regression Machines". In: Advances in Neural Information Processing Systems. Ed. by M. C. Mozer, M. I. Jordan, and T. Petsche. MIT Press, 1997, pp. 155–161.

[192] Qingyun Ge, Caimei Li, and Fulian Yang. "Support Vector Machine to Predict the Pile Settlement using Novel Optimization Algorithm". In: Geotechnical and Geological Engineering 41 (June 2023), pp. 1–15. DOI: 10.1007/s10706-023-02487-5.

[193] Sharif. Support Vector Regression Made Easy (with Python Code). Publication date not available. URL: https://www.aionlinecourse.com/tutorial/machine-learning/support-vector-regression.

[194] Fan Zhang and Lauren J. O'Donnell. "Chapter 7 - Support vector regression". In: Machine Learning. Ed. by Andrea Mechelli and Sandra Vieira. Academic Press, 2020, pp. 123–140. ISBN: 978-0-12-815739-8. DOI: https://doi.org/10.1016/B978-0-12-815739-8.00007-9. URL: https://www.sciencedirect.com/science/article/pii/B9780128157398000079.

[195] MathWorks. Understanding Support Vector Machine Regression. Publication date not available. URL: https://www.mathworks.com/help/stats/understanding-support-vector-machine-regression.html.

[196] Deepika Singh. Linear, Lasso, and Ridge Regression with scikit-learn. May 2019. URL: https://www.pluralsight.com/guides/linear-lasso-ridge-regression-scikit-learn/.

[197] Adityanarayanan Radhakrishnan. MLClassLecture2.pdf. Edited by: Max Ruiz Luyten, George Stefanakis, Cathy Cai. Jan. 2022. URL: https://web.mit.edu/modernml/course/lectures/MLClassLecture2.pdf.

[198] Mohit Baliyan. Implementation of Ridge Regression from Scratch using Python. Publication date not available. URL: https://www.geeksforgeeks.org/implementation-of-ridge-regression-from-scratch-using-python/.

[199] Wenwei Xu. What's the difference between Linear Regression, Lasso, Ridge, and ElasticNet in sklearn? Aug. 2019. URL: https://towardsdatascience.com/whats-the-difference-between-linear-regression-lasso-ridge-and-elasticnet-8f997c60cf29.

[200] GeeksforGeeks. ML – Different Regression types. July 2021. URL: https://www.geeksforgeeks.org/ml-different-regression-types/.

[201] Akash Kumar. Python | Implementation of Polynomial Regression. Publication date not available. URL: https://www.geeksforgeeks.org/python-implementation-of-polynomial-regression/.

[202] L. E. Eberly. "Multiple Linear Regression". In: Humana Press 2007. 2007. DOI: 10.1007/978-1-59745-530-5_9. URL: https://dx.doi.org/10.1007/978-1-59745-530-5_9.

[203] Penn State University. 7.7 - Polynomial Regression. Publication date not available. URL: https://online.stat.psu.edu/stat462/node/158/#:~:text=Although%20this%20model%20allows%20for,.%20.%20.%20%2C%20%CE%B2%20h%20!.

[204] Comite Europeen de Normalisation, ed. Bituminous Mixtures–Test Methods for Hot Mix Asphalt. Part 24; Resistance to Fatigue. 2007.

[205] Jayant Sirohi and Inderjit Chopra. "Fundamental Understanding of Piezoelectric Strain Sensors". In: Journal of Intelligent Material Systems and Structures 11.4 (2016), pp. 246–257. ISSN: 1045-389X 1530-8138. DOI: 10.1106/8bfb-gc8p-xq47-ycq0.

[206] Sanjay Yadav and Sanyam Shukla. "Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification". In: Institute of Electrical and Electronics Engineers Inc. 2016, pp. 78–83. ISBN: 9781467382861. DOI: 10.1109/IACC.2016.25.

[207] Jamal I. Daoud. "Multicollinearity and Regression Analysis". In: IOP Conf. Series: Journal of Physics: Conf. Series 949 (2017), p. 012009. DOI: 10.1088/1742-6596/949/1/012009.

[208] John D. Morris and Mary G. Lieberman. "Multicollinearity's Effect on Regression Prediction Accuracy with Real Data Structures". In: General Linear Model Journal 44.1 (2018).

[209] Omardonia. The Data Wizard's Guide to Preprocessing and Feature Engineering. URL: https://levelup.gitconnected.com/the-data-wizards-guide-to-preprocessing-and-feature-engineering-e44ac2268739.

[210] Carlos Vladimiro Gonzalez Zelaya. "Towards Explaining the Effects of Data Preprocessing on Machine Learning". In: 2019 IEEE 35th International Conference on Data Engineering (ICDE). 2375-026X/19/$31.00 ©2019 IEEE. IEEE. 2019, 2375–026X/19/$31.00 ©2019 IEEE. DOI: 10.1109/ICDE.2019.00245. eprint: 2375-026X/19/\$31.00©2019IEEE. URL: https://ieeexplore.ieee.org/document/8671695.

[211] Xu Chu et al. "Data Cleaning: Overview and Emerging Challenges". In: Proceedings of the 2016 ACM SIGMOD International Conference on Management of Data. ACM. San Francisco, CA, USA, June 2016. ISBN: 978-1-4503-3531-7. DOI: 10.1145/2882903.2912574.

[212] Probyto Data Science and Consulting Pvt. Ltd. Data Science for Business Professionals: A Practical Guide for Beginners. BPB, 2020, p. 542.

[213] M.T. Nguyen, H.J. Lee, and J. Baek. "Fatigue analysis of asphalt concrete under indirect tensile mode of loading using crack images". In: Journal of Testing and Evaluation 41.1 (2013), pp. 148–158.

[214] Heavy.AI. Feature Engineering. URL: https://www.heavy.ai/technical-glossary/feature-engineering (visited on 11/19/2022).

[215] S. Krishnan et al. Activeclean: Interactive data cleaning while learning convex loss models. 2015. URL: http://arxiv.org/pdf/1601.03797.pdf.

[216] J. Mahler et al. "Learning Accurate Kinematic Control of Cable-Driven Surgical Robots Using Data Cleaning and Gaussian Process Regression". In: CASE. 2014.

[217] S. Krishnan et al. "A Methodology for Learning, Analyzing, and Mitigating Social Influence Bias in Recommender Systems". In: RecSys. 2014.

[218] Data Integration. Accessed on Date of Access. 2023. URL: https://ai-ml-analytics.com/data-integration/.

[219] Mervyn Stone. "Cross-validatory Choice and Assessment of Statistical Predictions". In: Journal of the Royal Statistical Society: Series B (Methodological) 36.2 (1974), pp. 111–133.

[220] Brian S Everitt and Anders Skrondal. The Cambridge Dictionary of Statistics. 4th. Cambridge University Press, 2002.

[221] D. Maulud and A. M. Abdulazeez. "A Review on Linear Regression Comprehensive in Machine Learning". In: Journal of Applied Science and Technology Trends 1.4 (2020), pp. 140–147. DOI: 10.38094/jastt1457.

[222] Marko Sarstedt and Erik Mooi. A Concise Guide to Market Research. Vol. 12. Springer Berlin Heidelberg, 2014. ISBN: 978-3-662-56706-7. DOI: 10.1007/978-3-662-56707-4. URL: http://link.springer.com/10.1007/978-3-662-56707-4.

[223] Jason W. Osborne. "Prediction in Multiple Regression". In: Practical Assessment, Research, and Evaluation 7 (2000), Article 2. DOI: 10.7275/7j20-gg86. URL: https://scholarworks.umass.edu/pare/vol7/iss1/2.

[224] Marko Sarstedt, Erik Mooi, et al. A Concise Guide to Market Research. Vol. 12. Springer Berlin Heidelberg, 2014. ISBN: 978-3-662-56706-7. DOI: 10.1007/978-3-662-56707-4. URL: http://link.springer.com/10.1007/978-3-662-56707-4.

[225] Benyamin Ghojogh and Mark Crowley. "The Theory Behind Overfitting, Cross Validation, Regularization, Bagging, and Boosting: Tutorial". In: (2019). DOI: 10.48550/ARXIV.1905.12787. URL: https://arxiv.org/abs/1905.12787.

[226] Alan O. Sykes. "An Introduction to Regression Analysis". In: (1993). Coase-Sandor Institute for Law Economics Working Paper No. 20. URL: https://chicagounbound.uchicago.edu/law_and_economics.

[227] Multicollinearity. URL: https://datatab.net/tutorial/multicollinearity.

[228] John D. Morris and Mary G. Lieberman. "Multicollinearity's Effect on Regression Prediction Accuracy with Real Data Structures". In: General Linear Model Journal 44.1 (2018).

[229] Alan O Sykes. An Introduction to Regression Analysis. 1993. URL: https://chicagounbound.uchicago.edu/law_and_economics.

[230] Adam Hayes. Multiple Linear Regression (MLR) Definition, Formula, and Example. Updated April 29, 2023. Reviewed by Eric Estevez. Fact checked by Timothy Li. 2023. URL: https://www.investopedia.com/terms/m/mlr.asp.

[231] Qiao Dong et al. "Data Analysis in Pavement Engineering: An Overview". In: IEEE Transactions on Intelligent Transportation Systems (2021). ISSN: 15580016. DOI: 10.1109/TITS.2021.3115792.

[232] Joost Droogers. "Asphalt Concrete Stiffness Prediction Based on Composition and Binder Properties". 2018.

[233] Giulia Martini. "Predictive Modelling of Asphalt Concrete Functional Properties Using Multiple Linear Regression and Gradient Boosting". 2019.

[234] Scott M Lundberg and Su-In Lee. "A Unified Approach to Interpreting Model Predictions". In: Advances in Neural Information Processing Systems 30. 2017.

[235] Kasthurirangan Gopalakrishnan and Sunghwan Kim. "Support Vector Machines Approach to HMA Stiffness Prediction". In: (2011). DOI: 10.1061/ASCEEM.1943-7889.0000214.

[236] Dana Daneshvar and Ali Behnood. "Estimation of the Dynamic Modulus of Asphalt Concretes Using Random Forests Algorithm". In: International Journal of Pavement Engineering 23.2 (2022), pp. 250–260. ISSN: 1477268X. DOI: 10.1080/10298436.2020.1741587.

[237] Jia Wu, SenPeng Chen, and XiYuan Liu. "Efficient Hyperparameter Optimization through Model-Based Reinforcement Learning". In: Neurocomputing 409 (2020), pp. 381–393. ISSN: 18728286. DOI: 10.1016/j.neucom.2020.06.064.

[238] Li Yang and Abdallah Shami. "On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice". In: Neurocomputing 415 (2020), pp. 295–316. ISSN: 18728286. DOI: 10.1016/j.neucom.2020.07.061.

[239] James Bergstra, Daniel Yamins, and David Cox. "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures". In: International Conference on Machine Learning. 2013, pp. 115–123.

[240] Qi Wang et al. "A Comprehensive Survey of Loss Functions in Machine Learning". In: Annals of Data Science 9 (2022), pp. 187–212. ISSN: 21985812. DOI: 10.1007/s40745-020-00253-5.

[241] scikit-learn Contributors. sklearn.metrics.r2_score. Accessed on Date of access. September 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#sklearn.metrics.r2_score.

[242] scikit-learn Contributors. sklearn.metrics.mean_absolute_error. Accessed on Date of access. September 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html.

[243] scikit-learn Contributors. sklearn.metrics.mean_squared_error. Accessed on Date of access. September 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html.

[244] Timothy O Hodson. "Root-Mean-Square Error (RMSE) or Mean Absolute Error (MAE): When to Use Them or Not". In: Geoscientific Model Development 15 (2022), pp. 5481–5487. ISSN: 19919603. DOI: 10.5194/gmd-15-5481-2022.

[245] Linda Schulze Waltrup et al. "In: Statistical Modelling". In: 5 (), pp. 433–456. ISSN: 14770342. DOI: 10.1177/1471082X14561155.

[246] scikit-learn Contributors. sklearn.model_selection.cross_val_score. Accessed on Date of access. September 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html.

[247] Praveen Nellihela. What is K-fold Cross Validation? July 2022. URL: https://towardsdatascience.com/what-is-k-fold-cross-validation-5a7bb241d82f.

[248] scikit-learn contributors. Cross-validation: evaluating estimator performance. Accessed: 2023-09-17. URL: https://scikit-learn.org/stable/modules/cross_validation.html.

[249] Lloyd S Shapley. Notes on the n-Person Game – II: The Value of an n-Person Game. Tech. rep. RAND Corporation, 1951. URL: https://www.rand.org/content/dam/rand/pubs/research_memoranda/2008/RM670.pdf.

[250] S. Lundberg. SHAP. 2018. URL: https://github.com/slundberg/shap (visited on 11/04/2022).

[251] P.R. Edwards. Cumulative Damage in Fatigue with Particular Reference to the Effects of Residual Stresses. London: Her Majesty's Stationery Office, 1971.

[252] Yingjun Mei et al. "Fatigue Life Prediction of Asphalt Pavement Based on Cumulative Damage". In: Proceedings of the IEEE International Conference on Robotics and Automation. IEEE. 2010, pp. 739–749. DOI: 10.1109/ICRA.2010.5509833. URL: https://ieeexplore.ieee.org/document/5509833.

A

Pearson Correlation Matrix and Heat Map Diagram

The Pearson correlation formula assesses the relationship between variable x and variable y. The correlation matrix, in this context, forms a diagonal matrix, where the diagonal values are invariably set to 1. This arises from the fact that a variable is inherently correlated with itself, and, consequently, the correlation coefficient is always 1. Given that all the variables under consideration in this analysis represent voltages, the observed high correlation between them is indicative of multicollinearity, a phenomenon commonly encountered in such cases. It is important to note that multicollinearity, while present in this dataset, typically does not negatively impact regression and prediction tasks [228], given the nature of the variables involved. In the matrix below, a number from 0 to 9 means simply the variable so 0 is a list of all the voltages for frequency 15 Hz up to 9 which is a list of all the voltages for frequency of 35 Hz where each voltage is a response from the sensor for particular strain level.

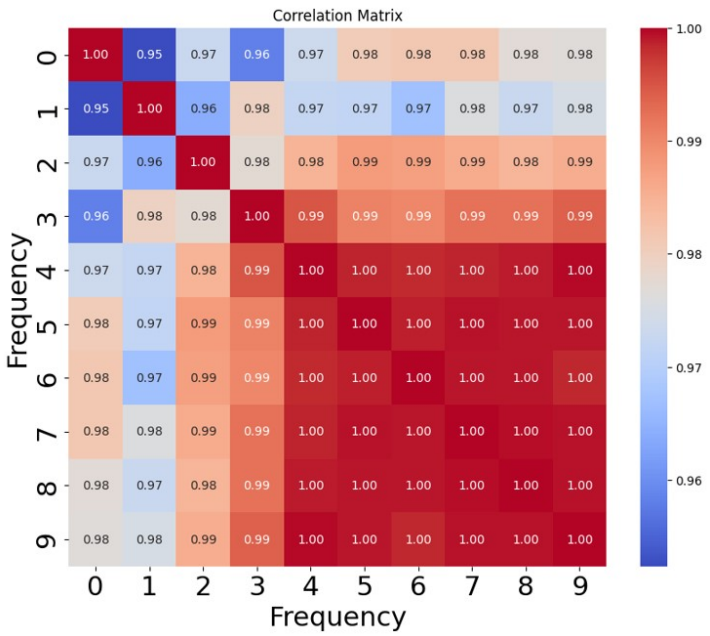


Figure A.1: Heat map diagram based on Pearson correlation matrix

B

Python Code (Machine Learning)

Importing necessary libraries

```
# Data manipulation and visualization
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from tabulate import tabulate  # For tabular data display
from IPython.display import display  # For enhanced data display in Jupyter notebooks

# File handling and I/O
import csv  # CSV file handling
import os   # Operating system-related functions

# Enum and data types
from enum import Enum  # Enumeration types
from typing import Union  # Type hints for function parameters

# Scientific computing and optimization
from scipy.optimize import curve_fit  # Curve fitting
import numpy.typing  # Typing for NumPy arrays
import rfcnt  # Real-time data collection

# Signal processing and analysis
from scipy.signal import argrelextrema, savgol_filter  # Signal processing functions

# Machine learning
from sklearn.linear_model import LinearRegression  # Linear regression model
from sklearn.metrics import r2_score  # R-squared score for model evaluation

# Scientific computation and statistics
from scipy import optimize  # General-purpose optimization
from scipy.stats import linregress  # Linear regression and statistical analysis

import time  # Time for timing code execution
import seaborn as sns  # Seaborn for enhanced data visualization

# Libraries for model selection and evaluation
from sklearn.model_selection import (
    train_test_split, cross_val_score, GridSearchCV
```

```
)  # Scikit-Learn for model selection, evaluation, and hyperparameter tuning
from sklearn.metrics import (
    mean_squared_error, mean_absolute_error, r2_score, make_scorer
)  # Scikit-Learn for common regression metrics

# Libraries for various machine learning models
# Scikit-Learn for linear regression models
from sklearn.linear_model import (
    LinearRegression, Ridge, Lasso, ElasticNet
)
# Scikit-Learn for polynomial features
from sklearn.preprocessing import PolynomialFeatures
from sklearn.svm import SVR  # Scikit-Learn for Support Vector Regression
# Scikit-Learn for K-Nearest Neighbors Regression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import (
    RandomForestRegressor, GradientBoostingRegressor,
    ExtraTreesRegressor, BaggingRegressor
)  # Scikit-Learn for ensemble models (Random Forest, Gradient Boosting, etc.)
 # Scikit-Learn for Decision Tree Regression
from sklearn.tree import DecisionTreeRegressor
 # Scikit-Learn for Multi-Layer Perceptron (Neural Network)
from sklearn.neural_network import MLPRegressor Regression
from prettytable import PrettyTable  # PrettyTable for creating tabular data displays

# Library for SHAP explanations and visualization
import shap  # SHAP for explaining and visualizing model predictions


```

Preparing data and splitting

```
# Define the dataset (voltage measurements for different frequencies)
ch1_values_freq_1_Hz
ch1_values_freq_2_Hz
ch1_values_freq_2_5_Hz
ch1_values_freq_5_Hz
ch1_values_freq_10_Hz
ch1_values_freq_15_Hz
ch1_values_freq_20_Hz
ch1_values_freq_25_Hz
ch1_values_freq_30_Hz
ch1_values_freq_35_Hz

# Define the corresponding strain values
strain_values = [15, 50, 100, 150, 200, 250, 300, 350, 400, 450]

# Prepare the data
voltages = []  # List to store voltage measurements
frequencies = []  # List to store corresponding frequencies
strains = []  # List to store corresponding strain values

# Define the list of frequencies
freqs = [1, 2, 2.5, 5, 10, 15, 20, 25, 30, 35]

# Define a list of voltage measurements for each frequency
ch_values = [ch1_values_freq_1_Hz, ch1_values_freq_2_Hz, ch1_values_freq_2_5_Hz,
```

```
            ch1_values_freq_5_Hz, ch1_values_freq_10_Hz, ch1_values_freq_15_Hz,
            ch1_values_freq_20_Hz, ch1_values_freq_25_Hz, ch1_values_freq_30_Hz,
            ch1_values_freq_35_Hz]

# Populate the voltages, frequencies, and strains lists
for freq, ch_values_freq in zip(freqs, ch_values):
    voltages.extend(ch_values_freq)
    frequencies.extend([freq] * len(ch_values_freq))
    strains.extend(strain_values)

# Split the data into training and testing sets
X = np.array(list(zip(voltages, frequencies))  # Feature matrix (voltage and frequency)
y = np.array(strains)  # Target values (strain)

# Split the data into training and testing sets using train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=30)

```

Functions

```
    def quantile_loss(y, y_pred, alpha=0.5):
    """
    Calculate the quantile loss between true and predicted values.

    Parameters:
        y (array-like): True target values.
        y_pred (array-like): Predicted target values.
        alpha (float, optional): Quantile level. Defaults to 0.5 (median).

    Returns:
        float: The mean quantile loss.
    """
    # Calculate the residuals
    residuals = y - y_pred

    # Calculate the quantile loss
    loss = np.sum(np.where(residuals >= 0, alpha * residuals, (alpha - 1) * residuals))

    # Calculate the mean loss
    mean_loss = np.mean(loss)

    return mean_loss

def expectile_loss(y, y_pred, alpha=0.5):
    """
    Calculate the expectile loss between true and predicted values.

    Parameters:
        y (array-like): True target values.
        y_pred (array-like): Predicted target values.
        alpha (float, optional): Expectile level. Defaults to 0.5 (median).

    Returns:
        float: The mean expectile loss.
    """
    # Calculate the residuals
    residuals = y - y_pred
```

```
38
39      # Calculate the expectile loss
40      loss = np.sum(np.where(residuals >= 0, (1 - alpha) * residuals**2, alpha * residuals**2))
41
42      # Calculate the mean loss
43      mean_loss = np.mean(loss)
44
45      return mean_loss
46
47  def neg_root_mean_squared_error(y_true, y_pred):
48      """
49      Calculate the negative root mean squared error (RMSE) between true and predicted values.
50
51      Parameters:
52          y_true (array-like): True target values.
53          y_pred (array-like): Predicted target values.
54
55      Returns:
56          float: Negative RMSE value.
57      """
58      mse = mean_squared_error(y_true, y_pred)
59      return -np.sqrt(mse)
60
61
62  def calculate_scores(model, X_test, y_test):
63
64      """
65      Given a model, this function calculates RMSE, R2, and MAE on the test data.
66      param model: A machine learning model.
67      param X_test: The test data features.
68      param y_test: The actual target values in the test data.
69      return: A tuple containing RMSE, R2, and MAE scores.
70
71
72      Make predictions on the test data using the provided model.
73      """
74      y_pred = model.predict(X_test)
75
76      """
77      Calculate the Root Mean Squared Error (RMSE).
78      """
79      rmse = np.sqrt(mean_squared_error(y_test, y_pred))
80
81      """
82      Calculate the R-squared (R2) score.
83      """
84      r2 = r2_score(y_test, y_pred)
85
86      """
87      Calculate the Mean Absolute Error (MAE).
88      """
89      mae = mean_absolute_error(y_test, y_pred)
90
91      """
92      Return the calculated RMSE, R2, and MAE as a tuple.
93      """
```

```
94      return rmse, r2, made
95
96  # Function to calculate damage based on Miner's rule
97  def calculate_damage(N, Nf, alpha, beta, C=1, epsilon_a=1):
98      return (C * (1 - alpha) / (1 + beta)) * (epsilon_a ** (beta + 1) * N) ** (1 / (1 - alpha))
99
100 # Function to fit damage parameters alpha and beta
101 def fit_damage_parameters(Nf, D_Nf, D_0, C=1):
102     def fitting_function(N, alpha, beta):
103         return calculate_damage(N, Nf, alpha, beta, C)
104
105     N_data = np.array([Nf, 0])
106     D_data = np.array([D_Nf, D_0])
107     p0 = [0.5, 0.5]
108     fitted_parameters, _ = curve_fit(fitting_function, N_data, D_data, p0=p0, maxfev=10000)
109
110     return fitted_parameters[0], fitted_parameters[1]
111
112     #Functions for importing and visualizing sensor data
113
114     import pandas as pd
115
116 def read_csv_file(filename, max_rows=None):
117     """
118     Read data from a CSV file and perform data preprocessing.
119
120     Parameters:
121         filename (str): The name of the CSV file to be read.
122         max_rows (int, optional): The maximum number of rows to read.
123
124     Returns:
125         numpy.ndarray: Transposed data from the CSV file.
126     """
127     # Construct the full file path based on the given filename
128     full_filename = "{filepath censored for pricacy}" + filename + ".csv"
129
130     # Read the CSV file, skip the first 12 rows, and select columns 0, 1, 2, and 3.
131     df = pd.read_csv(full_filename, skiprows=12, usecols=[0, 1, 2, 3],
132     names=['Source', 'CH1', 'CH2', 'CH3'], nrows=max_rows, low_memory=False)
133
134     # Drop rows with missing values in columns CH1, CH2, and CH3
135     df = df.dropna(subset=['CH1', 'CH2', 'CH3'])
136
137     # Convert columns Source, CH1, CH2, CH3 to numeric values and handle any conversion errors by
138     df[['Source', 'CH1', 'CH2', 'CH3']] = df[['Source', 'CH1', 'CH2', 'CH3']].apply
139     (pd.to_numeric, errors='coerce')
140
141     # Drop rows with NaN values
142     df = df.dropna()
143
144     # Return the transposed values as a NumPy array
145     return df.values.transpose()
146
147 def get_modified_first_column_and_CH1(filename):
148     """
149     Get the modified first column and CH1 data from a CSV file.
```

```
150
151     Parameters:
152         filename (str): The name of the CSV file to be read.
153
154     Returns:
155         numpy.ndarray: Modified first column and CH1 data.
156     """
157     # Read the data from the CSV file
158     data = read_csv_file(filename)
159
160     # Compute the modified first column as the time difference from the first data point
161     modified_first_column = data[0] - data[0][0]
162
163     # Extract CH1 data
164     ch1_data = data[1]
165
166     # To save unnecessary computation time, limit the data to the first 150,000 rows
167     return modified_first_column[:150000], ch1_data[:150000]
168
169
```

Fitting and hyperparameters tuning

```
1       #Models for tuning hyperparameters
2   # Define a dictionary of machine learning models
3   models = {
4       'Linear Regression': LinearRegression(),
5       'Ridge Regression': Ridge(),
6       'Lasso Regression': Lasso(),
7       'ElasticNet Regression': ElasticNet(),
8       'Polynomial Regression': make_pipeline(PolynomialFeatures(degree=2), LinearRegression()),
9       'SVR': SVR(),
10      'K-Nearest Neighbors': KNeighborsRegressor(),
11      'Random Forest': RandomForestRegressor(),
12      'Decision Tree': DecisionTreeRegressor(),
13      'Gradient Boosting Regression': GradientBoostingRegressor(),
14      'Extra Trees Regression': ExtraTreesRegressor(),
15      'Bagging Regression': BaggingRegressor(),
16      'Neural Network': MLPRegressor(),
17      'CatBoost Regressor': CatBoostRegressor()
18  }
19
20  # Define hyperparameter grids for Grid Search
21  param_grids = {
22      'CatBoost Regressor': {
23          # Hyperparameter grid for CatBoost Regressor
24          'iterations': list(np.linspace(2, 100, 50, dtype=int)),
25          'learning_rate': [0.01, 0.1, 0.2],
26          'depth': [6, 8, 10, 15],
27          'l2_leaf_reg': [1, 3, 5, 7, 9, 10],
28      },
29      'SVR': {
30          # Hyperparameter grid for Support Vector Regression (SVR)
31          'C': [0.1, 1, 10, 100, 500, 1000, 5000, 10000],  # Regularization parameter
32          'epsilon': [0.01, 0.1, 0.2, 0.5, 1],  # Epsilon in the epsilon-SVR model
33          'kernel': ['rbf'],  # Kernel type
34          'degree': [2, 3, 4],  # Degree of the polynomial kernel (for 'poly' kernel)
```

```
            'gamma': [0.1, 1, 2, 5, 10],  # Kernel coefficient for 'rbf', 'poly',
            and 'sigmoid' kernels
        },
        'ElasticNet Regression': {
            # Hyperparameter grid for ElasticNet Regression
            'alpha': list(np.linspace(0.01, 1, 20)),
            'l1_ratio': list(np.linspace(0.01, 1, 20))
        },
        'Polynomial Regression': {
            # Hyperparameter grid for Polynomial Regression
            'polynomialfeatures__degree': [1, 2, 3, 4, 5]
        },
        'Linear Regression': {},
        'Ridge Regression': {'alpha': list(np.linspace(0, 20, num=100)},
        'Lasso Regression': {'alpha': list(np.linspace(0, 20, num=100)},
        'Bagging Regression': {
            # Hyperparameter grid for Bagging Regression
            'n_estimators': list(np.linspace(2, 200, 100, dtype=int)),
            'max_samples': list(np.linspace(0.5, 1.0, 10)),
            'max_features': [0.1, 0.5, 0.75, 1.0, 2, 5],
        },
        'Decision Tree': {
            # Hyperparameter grid for Decision Tree
            'criterion': ['squared_error', 'friedman_mse', 'mae'],
            'splitter': ['best', 'random'],
            'max_depth': [None, 10, 20, 30, 50, 100, 200, 500],
            'min_samples_split': [2, 5, 10],
            'min_samples_leaf': [1, 2, 4, 8],
            'max_features': ['auto', 'sqrt', 'log2', None],
            'max_leaf_nodes': [None, 10, 20],
            'min_impurity_decrease': [0.0, 0.001, 0.1, 0.2],
            'min_weight_fraction_leaf': [0.0, 0.001, 0.1, 0.2],
        },
        'Gradient Boosting Regression': {
            # Hyperparameter grid for Gradient Boosting Regression
            'n_estimators': [50, 100, 200, 500],
            'max_depth': [3, 4, 5],
        },
        'Extra Trees Regression': {
            # Hyperparameter grid for Extra Trees Regression
            'n_estimators': [50, 100, 200],
            'max_depth': [None, 10, 20],
            'min_samples_split': [2, 5, 10],
            'bootstrap': [True, False],
            'warm_start': [True, False],
        },
        'Neural Network': {
            # Hyperparameter grid for Neural Network
            'hidden_layer_sizes': hidden_layer_sizes,
            'alpha': [0.0001, 0.001, 0.01, 0.1, 1, 2],
            'max_iter': [1000],
        },
        'Random Forest': {
            # Hyperparameter grid for Random Forest
            'n_estimators': [50, 100, 200, 400, 800, 1000, 1200, 1400, 1600, 1800, 2000],
            'max_features': ['auto', 'sqrt'],
```

```
        'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None],
        'min_samples_split': [2, 5, 10],
        'min_samples_leaf': [1, 2, 4, 6],
        'bootstrap': [True, False],
    },
    'K-Nearest Neighbors': {
        # Hyperparameter grid for K-Nearest Neighbors
        'n_neighbors': list(np.linspace(1, 200, 200, dtype=int)),
        'weights': ['uniform', 'distance'],
        'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
        'leaf_size': list(np.arange(1, 53, 2)),
        'p': [1, 2],
        'metric': ['euclidean', 'manhattan', 'chebyshev', 'minkowski']
    }
}

# Make it a scorer
neg_rmse_scorer = make_scorer(neg_root_mean_squared_error)


# Define a dictionary of loss functions, including Expectile Loss and R2
loss_functions = {
    'RMSE': make_scorer(mean_squared_error),
    'MAE': make_scorer(mean_absolute_error),
    'R2': make_scorer(r2_score),
    'Quantile Loss (0.5)': make_scorer(quantile_loss, quantile=0.5),
    'Expectile Loss (0.5)': make_scorer(expectile_loss, alpha=0.5),
}

# Create dictionaries to store best estimators, scores, and training times
best_estimators = {}
best_scores = {}
model_times = {}
best_hyperparameters = {}  # Store best hyperparameters


# Create a list to store model names and their corresponding times
model_times_list = []


# Iterate over models
for model_name, model in models.items():
    start_time = time.time()

    # Check if hyperparameter grid is defined for the current model
    if model_name in param_grids:
        param_grid = param_grids[model_name]

        # Create a GridSearchCV object for hyperparameter tuning
        grid_search = GridSearchCV(model, param_grid, cv=5, n_jobs=-1, scoring=neg_rmse_scorer)

        # Fit the GridSearchCV object to the training data
        grid_search.fit(X_train, y_train)

        # Get the best hyperparameters and the corresponding estimator
        best_params = grid_search.best_params_
```

```
        best_estimator = grid_search.best_estimator_

        # Store the best hyperparameters for this model
        best_hyperparameters[model_name] = best_params

        # Perform cross-validation with multiple loss functions
        cv = KFold(n_splits=5, shuffle=True, random_state=42)
        cv_scores = {}

        for loss_name, loss_func in loss_functions.items():
            # Calculate cross-validation scores using the selected loss function
            scores = cross_val_score(
                best_estimator, X_train, y_train, cv=cv, scoring=loss_func, n_jobs=-1
            )
            cv_scores[loss_name] = scores.mean()

        # Store the cross-validation scores for different loss functions
        cv_scores[loss_name] = scores.mean()

        # Store the best estimator and its corresponding R2 score (can be changed to
        other loss functions)
        best_estimators[model_name] = best_estimator
        best_scores[model_name] = cv_scores['R2']  # Using R2 for scoring

    end_time = time.time()

    # Calculate and store the time taken for hyperparameter tuning and cross-validation
    model_times[model_name] = end_time - start_time
    model_times_list.append([model_name, end_time - start_time])


# Print best hyperparameters for each model
for model_name, hyperparameters in best_hyperparameters.items():
    print(f"Best Hyperparameters for {model_name}:")
    for param, value in hyperparameters.items():
        print(f"{param}: {value}")

# Plot time vs model names
plt.figure(figsize=(13, 10))
plt.bar(model_times.keys(), model_times.values())
plt.xticks(rotation='vertical', fontsize=22)
plt.yticks(fontsize=22)
plt.xlabel('Model', fontsize=22)
plt.ylabel('Time (seconds)', fontsize=22)
plt.title('Model Training Time', fontsize=22)

# Set the y-axis limits to ensure at least 10 values are visible
plt.ylim(0, max(model_times.values()) + 2)  # Adjust the upper limit as needed

plt.tight_layout(h_pad=60)
plt.subplots_adjust(hspace=50)

plt.show()

# Display a table with model names and their corresponding times
print(tabulate(model_times_list, headers=["Model Name", "Time (seconds)"]))
```

Plotting RMSE, MAE and R2 scores

```
# Split your data into training and testing sets (adapt this part based on your data)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=30)

# Create dictionaries to store scores
rmse_scores = {}  # RMSE scores
r2_scores = {}    # R2 scores
mae_scores = {}   # MAE scores

# Iterate over models and calculate scores
for model_name, model in models.items():
    # Fit the model
    model.fit(X_train, y_train)

    # Calculate scores using the defined function
    rmse, r2, mae = calculate_scores(model, X_test, y_test)

    # Store scores in dictionaries
    rmse_scores[model_name] = rmse
    r2_scores[model_name] = r2
    mae_scores[model_name] = mae

# Create dictionaries for loss values
loss_values_rmse = [rmse_scores[model_name] for model_name in models.keys()]
loss_values_r2 = [r2_scores[model_name] for model_name in models.keys()]
loss_values_mae = [mae_scores[model_name] for model_name in models.keys()]

# Function to plot loss values
# This function creates bar plots to compare different models based on a given loss metric.
def plot_loss(selected_loss_name, selected_loss_values):
    plt.figure(figsize=(12, 6))
    bars = plt.bar(models.keys(), selected_loss_values)
    plt.xticks(rotation='vertical', fontsize=22)
    plt.ylabel(selected_loss_name, fontsize=22)
    plt.yticks(fontsize=22)
    plt.grid(True, axis='y')

    if selected_loss_name == 'R2':
        plt.yticks(np.linspace(0, 1, 10))
    else:
        plt.yticks(np.linspace(plt.gca().get_yticks()[0], plt.gca().get_yticks()[-1], 10))

    plt.title(f'{selected_loss_name} vs Model', fontsize=22)

    for bar, score in zip(bars, selected_loss_values):
        plt.text(bar.get_x() + bar.get_width() / 2 - 0.15, score + 0.01, f'{score:.4f}',
        fontsize=12)

# Plot RMSE, R2, and MAE
plot_loss('RMSE', loss_values_rmse)
plt.show()

plot_loss('R2', loss_values_r2)
```

```
plt.show()

plot_loss('MAE', loss_values_mae)
plt.show()

# Create a PrettyTable to display results with the same scores
table = PrettyTable()
table.field_names = ["Model", "RMSE", "R2", "MAE"]

# Add data to PrettyTable
for model_name in models.keys():
    table.add_row([model_name, rmse_scores[model_name], r2_scores[model_name],
    mae_scores[model_name])

# Print the results table
print("Performance Metrics for Tuned Models:")
display(table)

\end{verbatim}

\textbf{Plotting Quantile and Expectile scores
}
\begin{verbatim}
    # Create dictionaries to store scores for Quantile and Expectile losses
quantile_scores = {}
expectile_scores = {}

# Iterate over models and calculate Quantile and Expectile losses
for model_name, model in models.items():
    # Fit the model
    model.fit(X_train, y_train)

    # Calculate Quantile and Expectile losses
     # Example: alpha=0.5 for median
    quantile_loss_value = quantile_loss(y_test, model.predict(X_test), alpha=0.5)
    # Example: alpha=0.5 for median
    expectile_loss_value = expectile_loss(y_test, model.predict(X_test), alpha=0.5)

    # Store scores in dictionaries
    quantile_scores[model_name] = quantile_loss_value
    expectile_scores[model_name] = expectile_loss_value

# Create dictionaries for Quantile and Expectile loss values
loss_values_quantile = [quantile_scores[model_name] for model_name in models.keys()]
loss_values_expectile = [expectile_scores[model_name] for model_name in models.keys()]

# Function to plot Quantile and Expectile loss values
def plot_quantile_expectile_loss(selected_loss_name, selected_loss_values):
    plt.figure(figsize=(12, 6))
    bars = plt.bar(models.keys(), selected_loss_values)
    plt.xticks(rotation='vertical', fontsize=22)  # Increase fontsize to 22
    plt.ylabel(selected_loss_name, fontsize=22)  # Increase fontsize to 22
    plt.yticks(fontsize=22)  # Increase fontsize to 22
    plt.grid(True, axis='y')

    if selected_loss_name == 'R2':
```

```
        plt.yticks(np.linspace(0, 1, 10))
    else:
        plt.yticks(np.linspace(plt.gca().get_yticks()[0], plt.gca().get_yticks()[-1], 10))

    plt.title(f'{selected_loss_name} vs Model', fontsize=22)  # Increase fontsize to 22

    for bar, score in zip(bars, selected_loss_values):
        plt.text(bar.get_x() + bar.get_width() / 2 - 0.15, score + 0.01, f'{score:.4f}',
        fontsize=12)  # Increase fontsize to 22

# Plot Quantile and Expectile losses
plot_quantile_expectile_loss('Quantile Loss (0.5)', loss_values_quantile)
plt.show()

plot_quantile_expectile_loss('Expectile Loss (0.5)', loss_values_expectile)
plt.show()

```

Plotting and calculating SHAP values

```
    # Define a dictionary of machine learning models
models = {
    'Linear Regression': LinearRegression(),
    # Ridge regression with a specific alpha value
    'Ridge Regression': Ridge(alpha=15.56),
    # Lasso regression with a specific alpha value
    'Lasso Regression': Lasso(alpha=8.69),
     # ElasticNet regression with specific alpha and l1_ratio
    'ElasticNet Regression': ElasticNet(alpha=0.27, l1_ratio=0.11),
    # Polynomial regression
    'Polynomial Regression': make_pipeline(PolynomialFeatures(degree=4), LinearRegression()),

    'SVR': SVR(C=5000, degree=2, epsilon=0.01, gamma=0.1, kernel='rbf'),
    # Support Vector Regressor with specific hyperparameters

    'K-Nearest Neighbors': KNeighborsRegressor(n_neighbors=2, p=1, weights='distance'),
    # K-Nearest Neighbors with specific settings

    'Decision Tree': DecisionTreeRegressor(max_depth=500, min_impurity_decrease=0.2),
    # Decision Tree with specific hyperparameters

    'Random Forest': RandomForestRegressor(n_estimators=100),
    # Random Forest with a specific number of trees

    'Gradient Boosting': GradientBoostingRegressor(max_depth=3, n_estimators=200),
    # Gradient Boosting with specific settings

    'Extra Trees': ExtraTreesRegressor(n_estimators=100),
    # Extra Trees with a specific number of trees

    'Bagging': BaggingRegressor(n_estimators=18),
    # Bagging with a specific number of base estimators

    'Neural Network': MLPRegressor(alpha=1, hidden_layer_sizes=(10, 10, 10, 10, 10),
    max_iter=1000)
    # Neural Network with specific hyperparameters
}
```

```

# Iterate over the models
for model_name, model in models.items():
    # Train the machine learning model
    model.fit(X_train, y_train)

    # Create a SHAP explainer for the model's predictions
    explainer = shap.KernelExplainer(model.predict, X_train)

    # Calculate SHAP values for the test data
    shap_values = explainer.shap_values(X_test)

    # Create a custom summary plot using SHAP
    shap.summary_plot(shap_values, X_test, feature_names=["Voltage", "Frequency"])

    # Save the summary plot to a file (optional)
    plt.savefig(f"{model_name}_summary_plot.png")

    # Display the summary plot
    plt.show()

```

Plotting data training split percentage vs time

```
    # Define the list of training percentages to use
train_data_percentages = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]

# Create an empty dictionary to store R2 scores for each model
r2_scores = {}

# Split the dataset into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=30)

# Loop through each model and calculate R2 scores
for model_name, model in models.items():
    r2_list = []  # Create an empty list to store R2 scores for different training percentages

    # Set maximum iterations for specific models
    if model_name in ['Neural Network']:
        model.max_iter = 10000  # Adjust the maximum number of iterations here

    for train_percent in train_data_percentages:
        # Calculate the number of samples to use for training
        num_samples = int(len(X_train) * train_percent)
        if num_samples > 0:
            X_train_subset = X_train[:num_samples]
            y_train_subset = y_train[:num_samples]

            # Fit the model on the subset of data
            model.fit(X_train_subset, y_train_subset)

            # Make predictions on the validation set
            y_pred = model.predict(X_val[:num_samples])  # Adjust the dimension here

            # Calculate the R2 score and store it
            r2 = r2_score(y_val[:num_samples], y_pred)  # Use the correct r2_score function
            r2_list.append(r2)
```

```
34
35      # Store the R2 scores for the current model in the dictionary
36      r2_scores[model_name] = r2_list
37
38  # Create a single plot for R2 scores vs. training data percentage
39  plt.figure(figsize=(16, 8))  # Increased figsize
40  for model_name, r2_list in r2_scores.items():
41      plt.plot(train_data_percentages[:len(r2_list)], r2_list, label=model_name)
42
43  plt.xlabel('Training Data Percentage', fontsize=22)
44  plt.ylabel('R2 Score', fontsize=22)
45  plt.title('R2 Score vs. Training Data Percentage for Different Models', fontsize=22)
46  plt.legend(fontsize=18, bbox_to_anchor=(1.05, 1), loc='upper left')
47  plt.grid(True)
48  plt.xticks(fontsize=22)
49  plt.yticks(np.arange(0, 1.1, 0.1), fontsize=22)  # Set y-axis range and divisions
50  plt.ylim(0, 1)  # Set y-axis range
51  plt.show()
52
53  \end{verbatim}
54
55  \textbf{Predicted strain vs actual strain}
56
57  \begin{Verbatim}[numbers=left,xleftmargin=5mm]
58
59  # Create dictionaries to store results for tuned models
60  tuned_model_predictions = {}
61
62  # Create dictionaries to store R2 values for train and test subsets
63  r2_scores = {'Model': [], 'R2 Train': [], 'R2 Test': []}
64
65  # Import the r2_score function with a different name to avoid conflicts
66  from sklearn.metrics import r2_score as r2_score_func
67
68  # Iterate over tuned models
69  for model_name, model in models.items():
70      # Fit the model on the entire training data
71      model.fit(X_train, y_train)
72
73      # Predict on both training and test data
74      y_train_pred = model.predict(X_train)
75      y_test_pred = model.predict(X_test)
76
77      # Store predictions
78      tuned_model_predictions[model_name] = {'train_pred': y_train_pred, 'test_pred':
79      y_test_pred}
80
81      # Calculate R2 values for both train and test subsets
82      r2_train_score = r2_score_func(y_train, y_train_pred)
83      r2_test_score = r2_score_func(y_test, y_test_pred)
84
85      # Append R2 values to the dictionary
86      r2_scores['Model'].append(model_name)
87      r2_scores['R2 Train'].append(r2_train_score)
88      r2_scores['R2 Test'].append(r2_test_score)
89
```

```
# Display R2 values
r2_table = PrettyTable()
r2_table.field_names = ["Model", "R2 Train", "R2 Test"]
for i in range(len(r2_scores['Model'])):
    r2_table.add_row([r2_scores['Model'][i], r2_scores['R2 Train'][i],
    r2_scores['R2 Test'][i])
print(r2_table)

# Create the prediction curve plots for each tuned model
for model_name, predictions in tuned_model_predictions.items():
    plt.figure(figsize=(8, 6))
    plt.scatter(y_test, predictions['test_pred'], c='b', label='Test Data')
    plt.scatter(y_train, predictions['train_pred'], c='r', label='Train Data', alpha=0.5)
    plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], 'k--', lw=2,
    label='Perfect Fit (y=x)')
    plt.xlabel('Actual Strain', fontsize=22)
    plt.ylabel('Predicted Strain', fontsize=22)
    plt.title(f'Predicted vs Actual Strain ({model_name})', fontsize=22)
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()
```

Fatigue life prediction models
Cheng model

```

# Constants
T = 20  # Temperature constant
RP = 0  # Response Point constant

# Define the model function Cheng
def model(strain, A, B, C, D):
    """
    Define a mathematical model with parameters A, B, C, and D.
    This function calculates a predicted value based on the given parameters.

    Parameters:
        strain (float): The input strain value.
        A (float): Parameter A.
        B (float): Parameter B.
        C (float): Parameter C.
        D (float): Parameter D.

    Returns:
        float: The predicted value.
    """
    return A * (1/strain)**B * np.exp(C*T + D*RP)

# Define the objective function to minimize (using squared differences)
def objective(params):
    """
    Define an objective function to minimize.
    This function calculates the sum of squared differences
    between predicted and actual values.

    Parameters:
```

```
        params (list): A list of parameters to optimize (A, B, C, D).

    Returns:
        float: The objective value to minimize.
    """
    A, B, C, D = params
    predicted_Nf = model(strain_values, A, B, C, D)
    differences = predicted_Nf - Nf1_values  # Calculate the differences
    return np.sum(differences ** 2)  # Return the sum of squared differences

# Initial guess for parameters: A=1, B=1, C=0, D=0
initial_params = [1, 1, 0, 0]

# Perform optimization using L-BFGS-B
result = minimize(objective, initial_params, method='L-BFGS-B')

# Extract fitted parameters
A_fit, B_fit, C_fit, D_fit = result.x

# Print the fitted parameters
print("Fitted Parameters:")
print("A:", A_fit)
print("B:", B_fit)
print("C:", C_fit)
print("D:", D_fit)


```

Asphalt Institute Model

```
    # Constants
E0 = 13000 * 10**6  # A constant value (E0)

# Define the model function
def model(epsilon, k1, k2, k3):
    """
    Define a mathematical model with parameters k1, k2, and k3.

    Parameters:
        epsilon (float): The input variable.
        k1 (float): Parameter k1.
        k2 (float): Parameter k2.
        k3 (float): Parameter k3.

    Returns:
        float: The predicted value based on the given parameters.
    """
    return k1 * epsilon**k2 * E0**k3

# Fit the model to the data
params, _ = curve_fit(model, data[:, 0], data[:, 1], p0=(1, 1, 1))
# The curve_fit function is used to estimate the parameters k1, k2,
# and k3 that best fit the data.
# data[:, 0] corresponds to the first column of the data (input variable epsilon).
# data[:, 1] corresponds to the second column of the data (Number of cycles until failure).
# p0=(1, 1, 1) provides an initial guess for the parameters.

```

```
# Extract fitted parameters
k1_fit, k2_fit, k3_fit = params

# Print the fitted parameters
print("Fitted Parameters:")
print("k1:", k1_fit)
print("k2:", k2_fit)
print("k3:", k3_fit)

```

Only one strain is given as an example since beams were tested under constant strain test.

Chahboch and Lesne

```
# Define parameters
Nf, alpha, beta = 42778, 0.6, 1.9

# Function to calculate damage
def calculate_damage(N, Nf, alpha, beta):
    return 1 - (1 - (N / Nf) ** (1 / (1 - alpha))) ** (1 / (beta - 1))

# Generate N values from 0 to Nf
N_values = np.arange(Nf + 1)

# Calculate damage values using the function
damage_values = calculate_damage(N_values, Nf, alpha, beta)

# Plot the damage vs. number of cycles
plt.plot(N_values, damage_values, label='Damage curve')
plt.axhline(1, color='k', linestyle='--', label='Limit')

# Find the failure point
failure_point_x = np.argmax(damage_values >= 1)
failure_point_y = damage_values[failure_point_x]
plt.plot(failure_point_x, failure_point_y, 'ro', markersize=8,
         label=f'Failure Point (Nf={failure_point_x}, D={failure_point_y:.2f})')

# Find the x value where y is 1
x_value_at_y_equal_to_1 = N_values[np.argmax(damage_values >= 1)]
print("x value where y is 1:", x_value_at_y_equal_to_1)

# Calculate percentage difference
percentage_difference = (abs(x_value_at_y_equal_to_1 - Nf) / Nf) * 100
print("Number of cycles until failure: Nf =", Nf)
print("% difference: ", percentage_difference)

# Define parameters for the second part
A, B, C, epsilon = 0.87, -2, 0.45, 220

# Calculate Nf_funcValues using the defined function
Nf_funcValues = [A * (1/epsilon)**B * (1 - D)**C for D in damage_values]

# Plot the second part
plt.plot(damage_values, Nf_funcValues, label='Fatigue life line')
plt.axvline(1, color='black', ls='--', label='x = Nf')
plt.axhline(0, color='black')
plt.xlabel('Cumulative damage', fontsize=17)
plt.ylabel('Number of cycles', fontsize=17)
```

```
plt.grid()
plt.legend()
plt.show()

```

Bodin et al. model

```

# Known parameters
Nf = 1
D_Nf = 0
D_0 = 42778 #No damage hence full life
C = 1

# Fit the parameters alpha and beta
alpha, beta = fit_damage_parameters(Nf, D_Nf, D_0, C)
print("Fitted parameters: alpha =", alpha, ", beta =", beta)

# Generate N values from 0 to Nf
N_values = np.arange(0, Nf + 1)
damage_values = calculate_damage(N_values, Nf, alpha, beta, C)

# Plot the damage vs. number of cycles
plt.plot(N_values, damage_values, label='Damage')
plt.scatter(Nf, D_Nf, color='red', label=f'Known Point (Nf={Nf}, D_Nf={D_Nf})')
plt.axhline(y=1, color='black', linestyle='dashed', label='Fitted Line (y=1)')
plt.xlabel('Number of Cycles (N)', fontsize=17)
plt.ylabel('Damage (D)', fontsize=17)
plt.xticks(fontsize=17)
plt.yticks(fontsize=17)
plt.title('Damage vs. Number of Cycles')
plt.grid(True)
plt.legend()
plt.show()

# Define the function for Ma et al.
def func(D, A, B, C, epsilon):
    return A * (1/epsilon) ** B * (1 - D) ** C

A, B, C, epsilon = 0.87, -2, 0.4, 220
Nf_funcValues = [func(D_i, A, B, C, epsilon) for D_i in damage_values]

# Plot the Ma et al. function
plt.plot(damage_values, Nf_funcValues, label='Fatigue life line')
plt.axvline(1, color='black', ls='--', label='x = Nf')
plt.axhline(0, color='black')
plt.ylabel('Number of Cycles (Nf)', fontsize=17)
plt.xlabel('Cumulative Damage (D)', fontsize=17)
plt.title('Damage vs. Number of Cycles')
plt.xticks(fontsize=17)
plt.yticks(fontsize=17)
plt.grid()
plt.legend()
plt.show()
```

C

Python Code (Data Visualization and Post Processing)

Importing and post processing data for just frequency 1Hz and different strain levels

```
1
2  # Create a figure with a specific size
3  plt.figure(figsize=[210/25, 8])
4
5  # Iterate through different filenames to plot data
6  # The filenames represent different experimental conditions
7  filenames = [
8      "SDS00156",
9      "SDS00147",
10     "SDS00148",
11     "SDS00149",
12     "SDS00150",
13     "SDS00151",
14     "SDS00152",
15     "SDS00153",
16     "SDS00154",
17     "SDS00155",
18 ]
19
20 for filename in filenames:
21     # Get modified first column and CH1 data for the current filename
22     modified_first_column, ch1_data = get_modified_first_column_and_CH1(filename)
23
24     # Shift x-values for different plots to avoid overlap
25     shifted_x_values = [x + 0.2 for x in modified_first_column]
26
27     # Different line styles are used for visualization (ls parameter)
28     # Label indicates the deformation rate in µm/m
29     if "SDS00156" in filename or "SDS00148" in filename or "SDS00149" in filename:
30         plt.plot(shifted_x_values, ch1_data, label=f'{filename} (µm/m)', ls='--')
31     else:
32         plt.plot(shifted_x_values, ch1_data, label=f'{filename} (µm/m)')
33
34 # Set the title and axis labels
35 plt.title('Frequency 1Hz', fontsize=19)
```

```
plt.xlabel('Time [sec]', fontsize=24)
plt.ylabel('Voltage [V]', fontsize=24)

# Adjust font sizes for ticks and legends
plt.xticks(fontsize=19)
plt.yticks(fontsize=19)

# Add a legend to the plot
plt.legend(fontsize=16)

# Display grid lines
plt.grid()

# Show the plot
plt.show()
```

Importing and post processing data for just strain level of 15 micrometer/meter and different frequency levels [Hz]

```

# Create a figure with a specific size
plt.figure(figsize=[210/25, 8])

# Define a list of filenames that represent different frequencies
filenames = [
    "SDS00156", "SDS00157", "SDS00167", "SDS00177", "SDS00187",
    "SDS00197", "SDS00207", "SDS00217", "SDS00233"
]

# Iterate through the filenames to plot data for different frequencies
for filename in filenames:
    # Get modified first column and CH1 data for the current filename
    modified_first_column, ch1_data = get_modified_first_column_and_CH1(filename)

    # Extract the frequency value from the filename (e.g., "SDS00156" represents 1 Hz)
    frequency = int(filename[3:])  # Remove the "SDS" part and convert to an integer

    # Plot the data with the frequency as the label
    plt.plot(modified_first_column, ch1_data, label=f'{frequency} Hz')

# Set the title and axis labels
plt.title('Strain 15 µm/m', fontsize=16)
plt.xlabel('Time [sec]', fontsize=24)
plt.ylabel('Voltage [V]', fontsize=24)

# Adjust font sizes for ticks and legend
plt.xticks(fontsize=19)
plt.yticks(fontsize=19)

# Add a legend to the plot
plt.legend(fontsize=16)

# Display grid lines
plt.grid()
```

```

# Show the plot
plt.show()


```

Finding minimum and maximum voltage from sensor data

```
    # Create variables to store minimum and maximum values for CH1, CH2, and CH3

# CH1 represents PZT (voltage [V])
# CH2 represents PVDF (voltage [V])
# CH3 represents LVDT (voltage [V])

# Loop through file numbers to process data from different files
# Define the file numbers and corresponding frequencies

frequency_mapping = {
    1: [156]+list(range(147,156)),
    2: list(range(157,167)),
    5: list(range(167,177)),
    10: list(range(177,187)),
    15: list(range(187,197)),
    20: list(range(197,207)),
    25: list(range(207,217)),
    30: [217,218]+list(range(225,233)),
    35: list(range(233,243))
}# Initialize the frequency lists
frequency_lists = {f: [] for f in frequency_mapping}

# Populate the frequency lists with minimum and maximum values
for frequency, file_numbers in frequency_mapping.items():
    for file_number in file_numbers:
        # Access the variables dynamically using their names
        CH1_min = globals()[f"min_CH1_{file_number}"]
        CH1_max = globals()[f"max_CH1_{file_number}"]
        CH2_min = globals()[f"min_CH2_{file_number}"]
        CH2_max = globals()[f"max_CH2_{file_number}"]
        CH3_min = globals()[f"min_CH3_{file_number}"]
        CH3_max = globals()[f"max_CH3_{file_number}"]

        # Append the values to the corresponding frequency list
        frequency_lists[frequency].append((CH1_min, CH1_max, CH2_min, CH2_max,
        CH3_min, CH3_max))  # Added CH3

# Print the frequency lists
for frequency, values in frequency_lists.items():
    print(f"Frequency {frequency}:")
    for i, (CH1_min, CH1_max, CH2_min, CH2_max, CH3_min, CH3_max) in enumerate(values):

        print(f"File {frequency_mapping[frequency][i]} - CH1 Min: {CH1_min}, CH1 Max: {CH1_max},
        CH2 Min: {CH2_min}, CH2 Max: {CH2_max}, CH3 Min: {CH3_min}, CH3 Max: {CH3_max}")


```

Plotting voltage amplitude vs time

Example of plotting and finding the voltage amplitude also known as peak-to-peak voltage vs

strain

```

#lingress library required

# Frequencies for data collection
frequencies = [already loaded in]]

# Strain values
strain_values = [already loaded in]

# Max and min data for all frequencies
max_data = [max_ch1_freq_1_Hz, max_ch1_freq_2_Hz, max_ch1_freq_5_Hz, max_ch1_freq_10_Hz,
max_ch1_freq_15_Hz, max_ch1_freq_20_Hz, max_ch1_freq_25_Hz, max_ch1_freq_30_Hz, max_ch1_freq_35_H
min_data = [min_ch1_freq_1_Hz, min_ch1_freq_2_Hz, min_ch1_freq_5_Hz, min_ch1_freq_10_Hz,
min_ch1_freq_15_Hz, min_ch1_freq_20_Hz, min_ch1_freq_25_Hz, min_ch1_freq_30_Hz, min_ch1_freq_35_H

# Define colors for each frequency
colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k', 'purple', 'orange']

# Create a figure for the plots
plt.figure(figsize=(12, 15))

# Plot max values for each frequency
plt.subplot(311)
plt.title("Max Values for CH1")
for i, freq in enumerate(frequencies):
    color = colors[i]
    plt.scatter(strain_values, max_data[i], marker='o', label=f"{freq} Hz", color=color)
    slope, intercept, _, _, _ = linregress(strain_values, max_data[i])
    plt.plot(strain_values, intercept + slope * np.array(strain_values), '--', color=color)
plt.xlabel("Strain Values")
plt.ylabel("Max Values")
plt.legend()
plt.grid(True)

# Plot min values for each frequency
plt.subplot(312)
plt.title("Min Values for CH1")
for i, freq in enumerate(frequencies):
    color = colors[i]
    plt.scatter(strain_values, min_data[i], marker='o', label=f"{freq} Hz", color=color)
    slope, intercept, _, _, _ = linregress(strain_values, min_data[i])
    plt.plot(strain_values, intercept + slope * np.array(strain_values), '--', color=color)
plt.xlabel("Strain Values")
plt.ylabel("Min Values")
plt.legend()
plt.grid(True)

# Plot max + absolute(min) values for each frequency
plt.subplot(313)
plt.title("Max + |Min| Values for CH1")
for i, freq in enumerate(frequencies):
    color = colors[i]
    max_plus_abs_min = [max_val + abs(min_val) for max_val, min_val in zip(max_data[i],
    min_data[i])]
    plt.scatter(strain_values, max_plus_abs_min, marker='o', label=f"{freq} Hz", color=color)
```

```
    slope, intercept, _, _, _ = linregress(strain_values, max_plus_abs_min)
    plt.plot(strain_values, intercept + slope * np.array(strain_values), '--', color=color)
plt.xlabel("Strain Values")
plt.ylabel("Max + |Min| Values")
plt.legend()
plt.grid(True)

# Ensure tight layout for subplots
plt.tight_layout()
plt.show()

```

Python code for integral voltage vs time is attached as Appendix D.

D

Integral Voltage vs Strain

This appendix presents comprehensive data concerning the integral voltage versus strain. Initially, the voltage curves (voltage vs time) that result from sinusoidal loading (constant strain) are simplified by representing them as sinusoidal curves, characterized by their peak-to-peak voltage (voltage amplitude) and frequency. The following figures depict these curves:

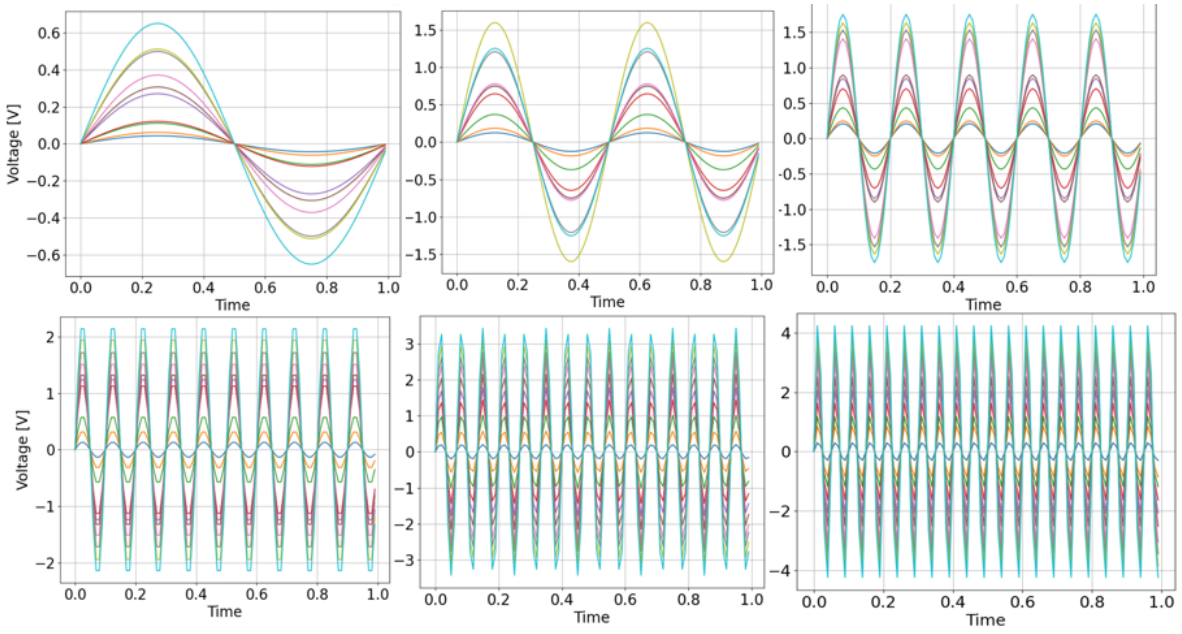


Figure D.1: Voltage vs time curve

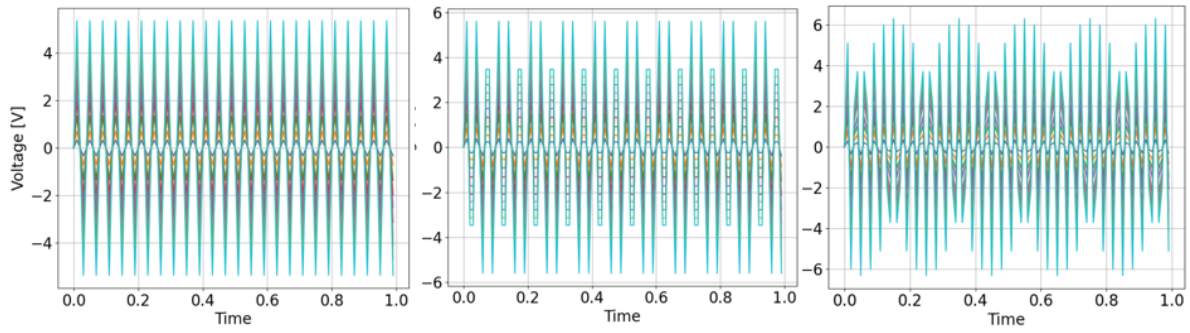


Figure D.2: Voltage vs time curve (continue)

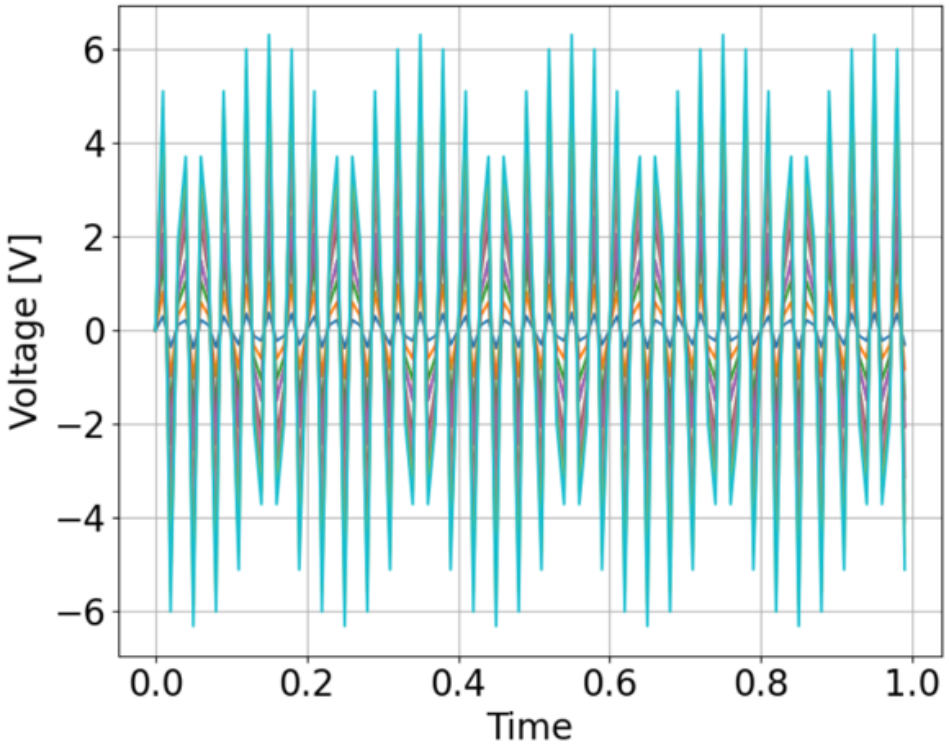


Figure D.3: Voltage vs time curve (continue)

Finally, Integral voltage (Vs) vs time is illustrated in the figure below. Integral voltage at strain of 200 and 35 Hz is an experimental error.

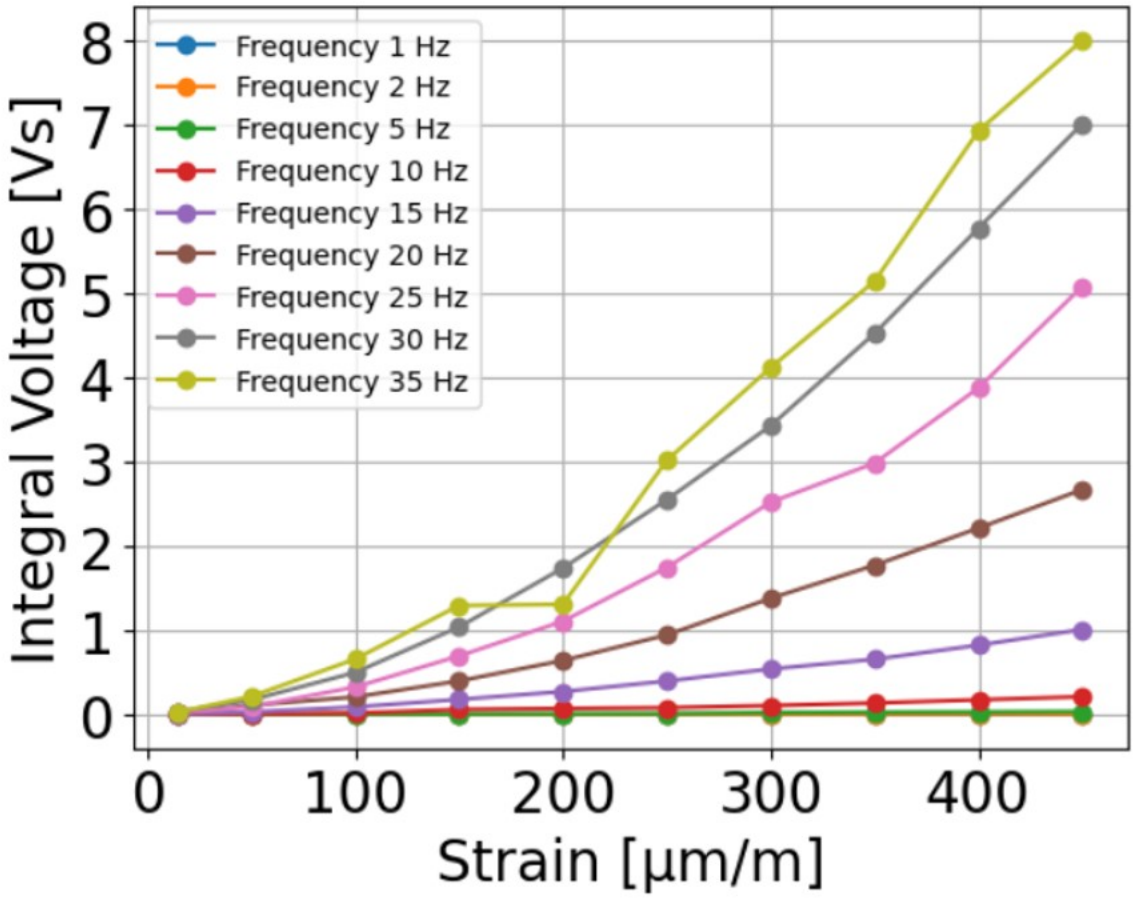


Figure D.4: Integral voltage Amplitude vs maximum strain curve

Python code used to calculate integral values and then to plot integral voltage [Vs] vs strain is given in the code below.

Integral voltage method

```

# Frequency values
frequencies = [frequency list]

# Amplitude peak-to-peak values for each frequency
ch1_values_freq = [
    # List of amplitude values for each frequency
]

# Create subplots with one subplot per row
num_rows = len(frequencies)
fig, axes = plt.subplots(num_rows, 1, figsize=(8, 6*num_rows), sharex=True)

# Time values (assuming it starts from 0 and increases by 0.1)
time_values = np.arange(0, 1, 0.01)

# Lists to store the areas under the curve and integral voltage values
areas_under_curve = []
integral_voltage_list = []
```

```
21
22  # Loop through each frequency and plot sinusoidal functions on corresponding subplots
23  for i, freq in enumerate(frequencies):
24      amplitude_values_pp = ch1_values_freq[i]
25      amplitude_values = [amp / 2 for amp in amplitude_values_pp]
26
27      # Lists to store areas and integral voltage values for this frequency
28      areas_freq = []
29      integral_voltage_freq = []
30
31      # Loop through each amplitude value and plot sinusoidal functions
32      for amplitude in amplitude_values:
33          sinus_func = amplitude * np.sin(2 * np.pi * freq * time_values)
34          axes[i].plot(time_values, sinus_func)
35
36          # Calculate the area under the curve using the trapezoidal rule
37          area = np.trapz(sinus_func, dx=0.01)
38          areas_freq.append(area)
39
40          # Calculate the integral voltage for this amplitude
41          integral_voltage = np.sqrt(2) * amplitude * freq * area
42          integral_voltage_freq.append(integral_voltage)
43
44      # Store the areas and integral voltage values for this frequency
45      areas_under_curve.append(areas_freq)
46      integral_voltage_list.append(integral_voltage_freq)
47
48      axes[i].set_title(f"Frequency {freq} Hz")
49      axes[i].set_ylabel("Amplitude")
50      axes[i].grid(True)
51
52  # Label the x-axis of the last subplot
53  axes[-1].set_xlabel("Time")
54
55  # Adjust layout to prevent overlapping
56  plt.tight_layout()
57
58  # Show the plot
59  plt.show()
60
61  # Plot the integral voltage vs strain for all frequencies
62  for i, freq in enumerate(frequencies):
63      plt.plot(strains, integral_voltage_list[i], 'o-', label=f"Frequency {freq} Hz")
64
65  plt.xlabel("Strain [µm/m]")
66  plt.ylabel("Integral Voltage [Vs]")
67  plt.title("Integral Voltage vs Strain for Different Frequencies")
68  plt.legend()
69  plt.grid(True)
70  plt.show()
71
```

E

PVDF

Within this appendix, diagrams pertaining to PVDF are provided for reference. The inclusion of this section in the appendix is necessitated by the fact that the data derived from PVDF is non-representative and cannot be effectively utilized due to the prevalence of noise, which surpasses the generated voltage.

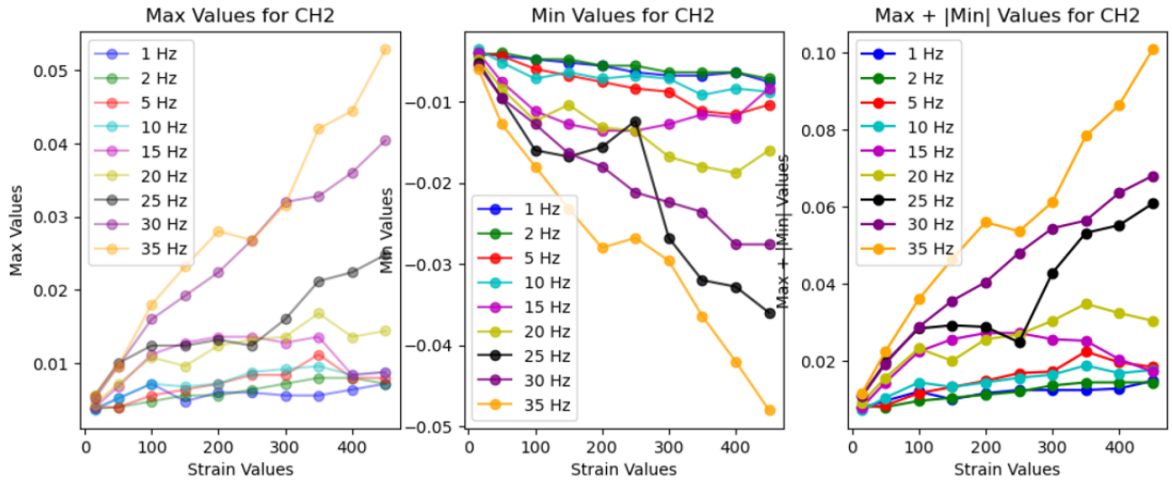


Figure E.1: Minimum, maximum and peak-to-peak PVDF voltage vs strain curves

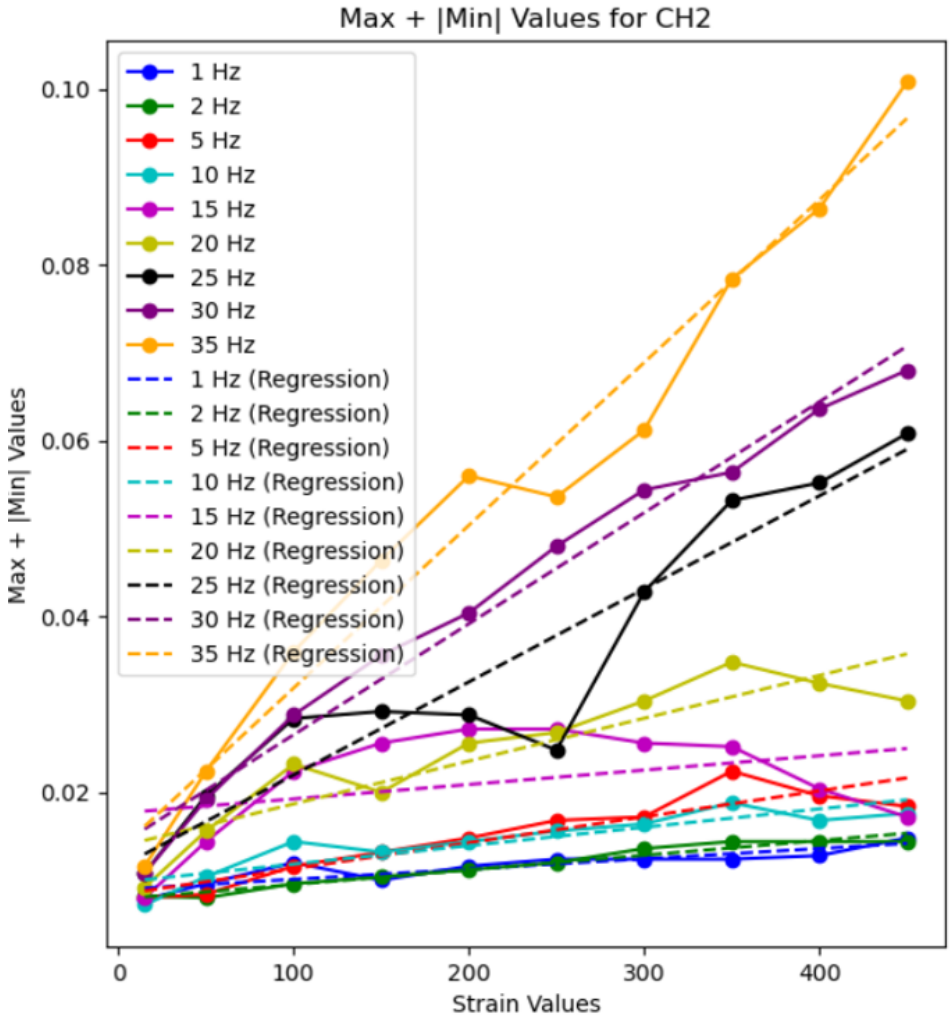


Figure E.2: PVDF voltage amplitude vs strain curve

F

COMSOL simulation results

Figure F.1 depicts a standard contour plot. The color variations exhibited in Figure F.1 are not uniform but the transition from red to blue. This color variance represents the sensor's flexing behavior during a simulated four-point bending process. As the sensor experiences deformation, the lower side elongates while the upper side contracts, and subsequently, the upper side elongates while the lower side contracts, as further illustrated in Section F.1. The specific position on the sensor determines the contour color.

Figures F.2 and F.3 present contour plots of the generated voltages as the strain level increases from 15 $[\frac{\mu m}{m}]$ to 400 $[\frac{\mu m}{m}]$. The corresponding data values have been tabulated and are included in Chapter 4, Results section 4.2. Screenshots of the original tables using Python are provided in Figure F.4a and F.4b. Due to the similarity in contour plots for frequencies of 20 [Hz] and 30 [Hz], only the tables generated by Python, accompanied by the percentage error, are included. It is worth noting that, at 30 Hz, there was an experimental defect in the data, as depicted in Figure 4.13a. As a result in figures F.5a and F.5b, the percentage difference is also shown for the 20 [Hz] and 30 [Hz] frequencies.

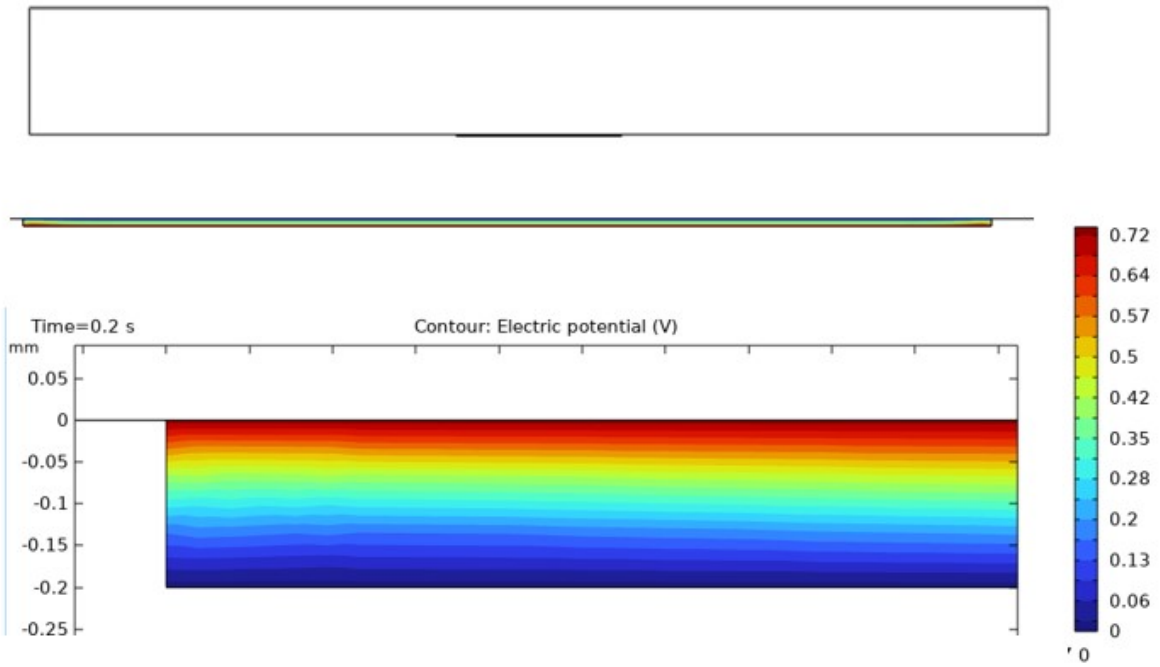


Figure F.1: Typical sensor contour

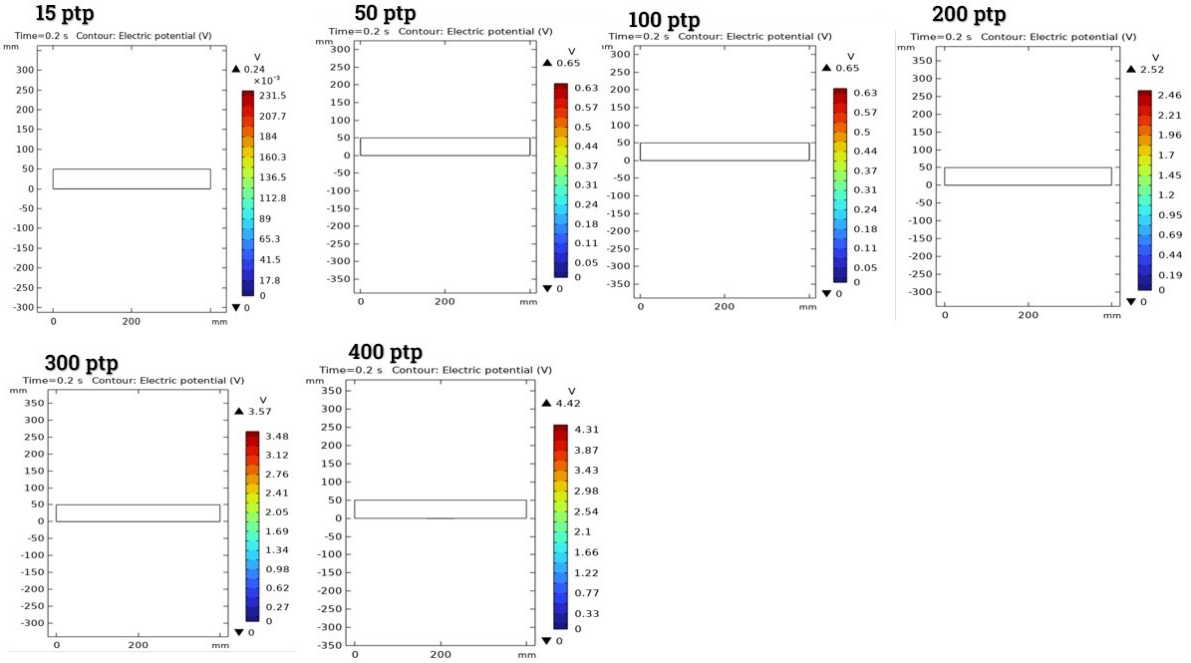


Figure F.2: COMSOL results for different strain levels and same frequency of 5 Hz

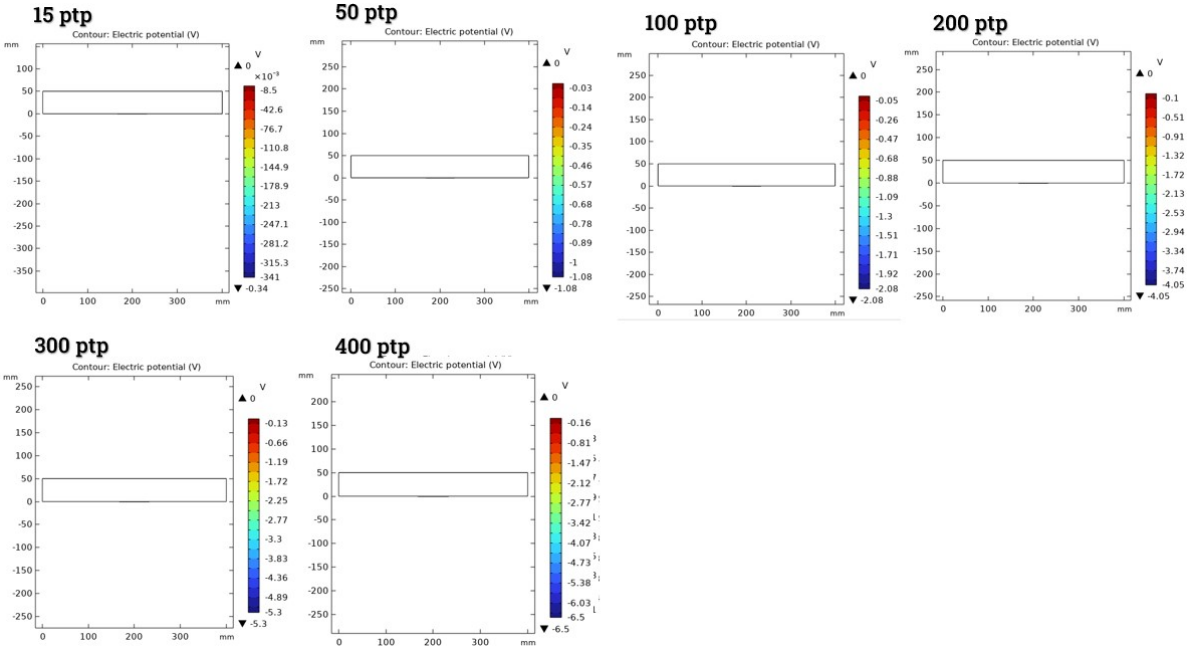


Figure F.3: COMSOL results for different strain levels and same frequency of 10 Hz

Strain	ch1_values_freq_5_Hz	Voltage_comsol_5Hz
15	0.2655	0.24
50	0.66	0.65
100	1.228	1.3
200	2.595	2.52
300	3.275	3.57
400	4.17	4.42

(a) COMSOL results for different strain levels and the same frequency of 5 Hz

Strain	ch1_values_freq_10_Hz	Voltage_comsol_10Hz
15	0.384	0.34
50	1.128	1.08
100	2.05	2.08
200	3.645	4.05
300	5.13	5.3
400	6.34	6.5

(b) COMSOL results for different strain levels and the same frequency of 10 Hz

Figure F.4: Comparison of PZT voltage amplitude from COMSOL and 4PB test for frequency of 5[Hz] and 10[Hz]

Strain	ch1_values_freq_20_Hz	Voltage_comsol_20Hz	%error
15	0.646	0.65	0.619195
50	1.53	1.5	-1.96079
100	2.78	2.75	-1.07914
200	5.01	5.24	4.59082
300	7.59	7.75	2.10804
400	9.52	9.6	0.840336

(a) COMSOL results for different strain levels and the same frequency of 20 Hz

Strain	ch1_values_freq_30_Hz	Voltage_comsol_30Hz	%error
15	0.746	0.74	-0.80429
50	2.125	2.1	-1.17647
100	3.65	3.71	1.64384
200	5.43	5.32	-2.02578
300	8.92	8.71	-2.35426
400	11.52	11.82	2.60417

(b) COMSOL results for different strain levels and the same frequency of 30 Hz

Figure F.5: Comparison of PZT voltage amplitude from COMSOL and 4PB test for frequency of 20[Hz] and 30[Hz]

G

COMSOL beam deflection

Figures below show how the beam is deflected in COMSOL FEA program to generate piezoelectric voltage as found in the section 4.2. It can be seen from beam deflection that it behaves as a real four-point bending test setup (4PB). This means that the beam is bent from its neutral position where it is completely horizontal to maximum deflection and then it comes back to its neutral position and goes in the opposite direction until deflection is again reached to its maximum.

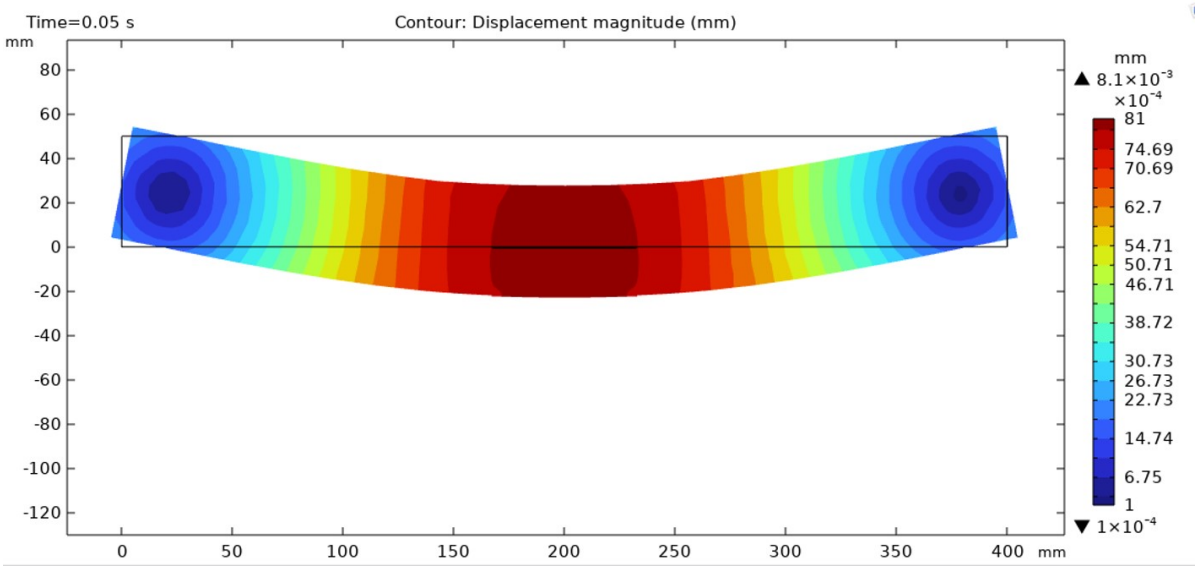


Figure G.1: COMSOL deflection for a strain of 15 $\frac{\mu m}{m}$

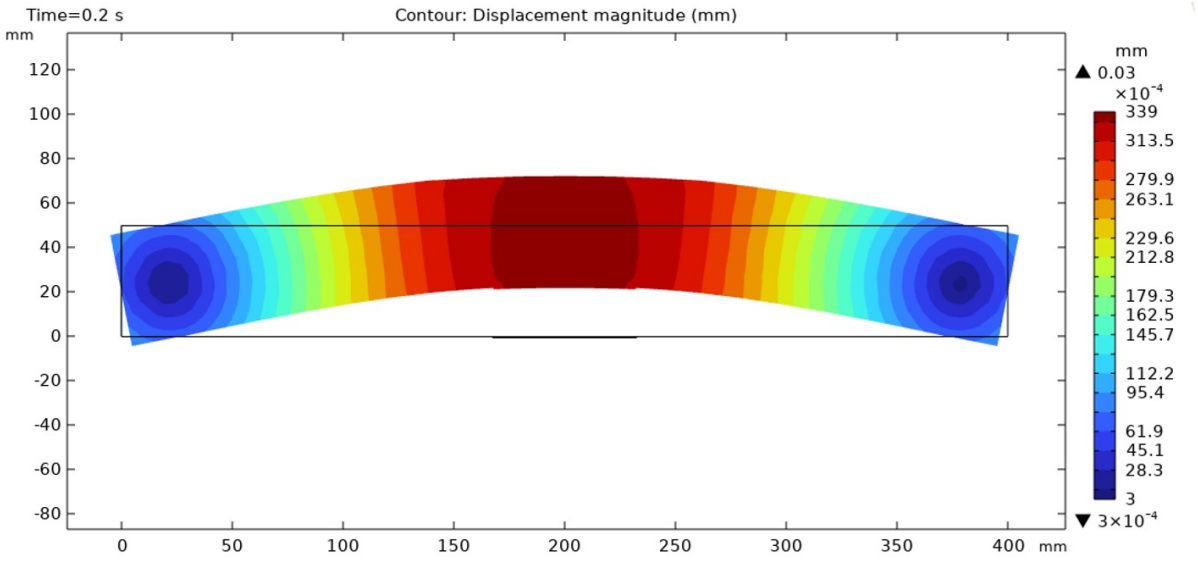


Figure G.2: COMSOL deflection for a strain of 50 $\frac{\mu m}{m}$

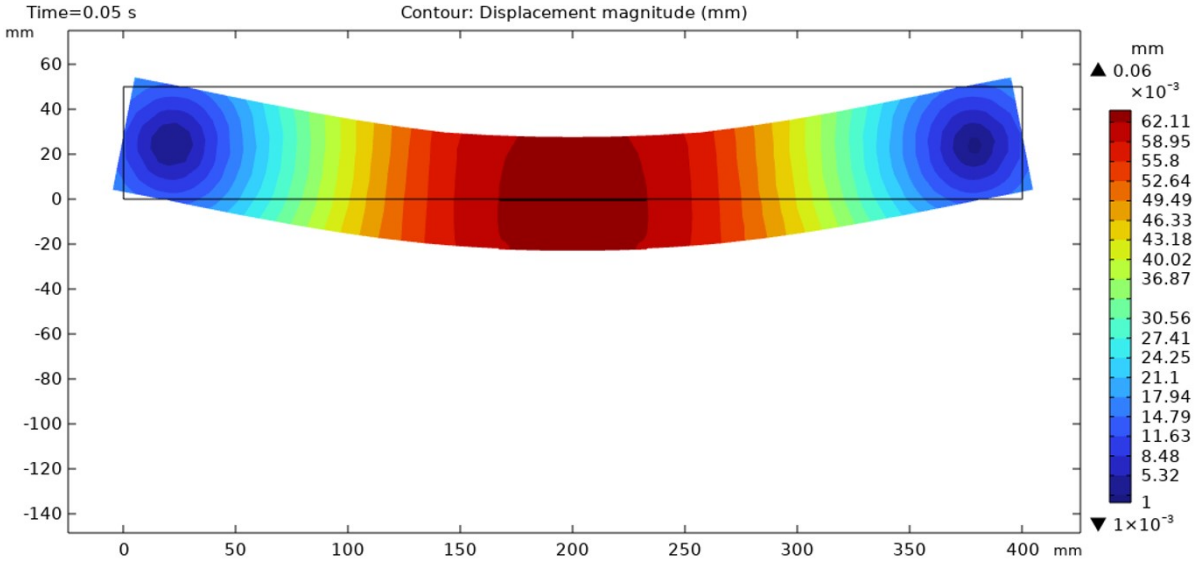


Figure G.3: COMSOL deflection for a strain of 100 $\frac{\mu m}{m}$

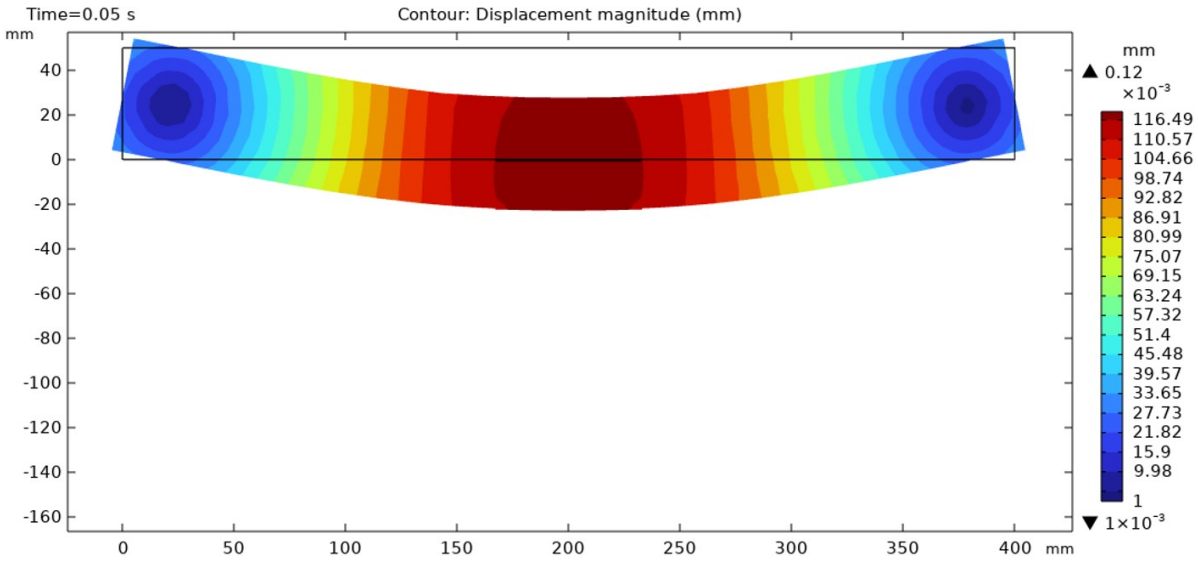


Figure G.4: COMSOL deflection for a strain of 200 $\frac{\mu m}{m}$

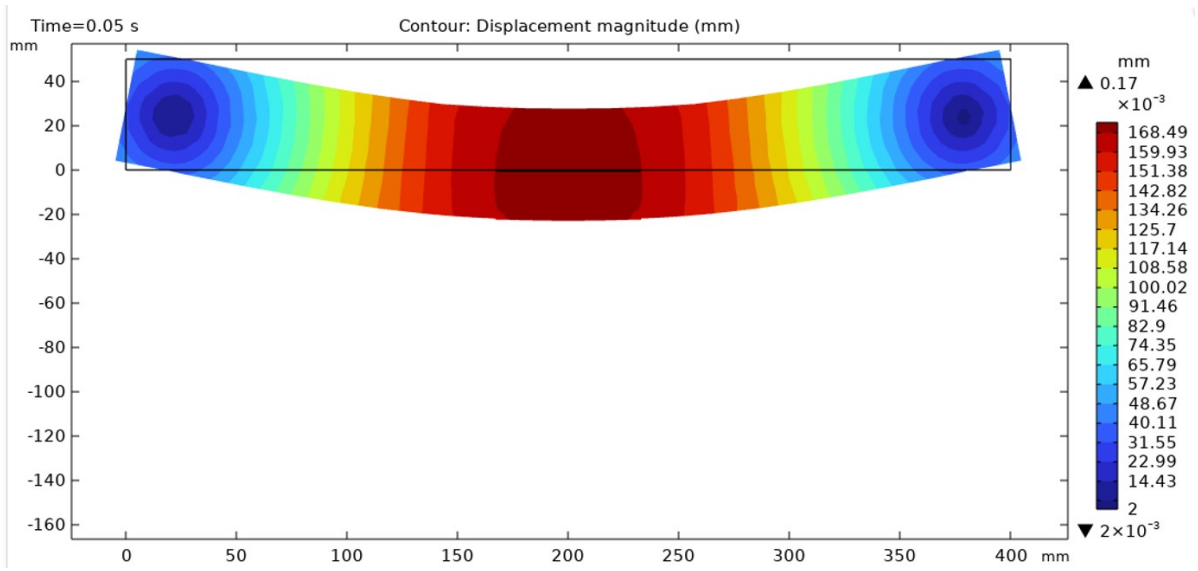


Figure G.5: COMSOL deflection for a strain of 300 $\frac{\mu m}{m}$

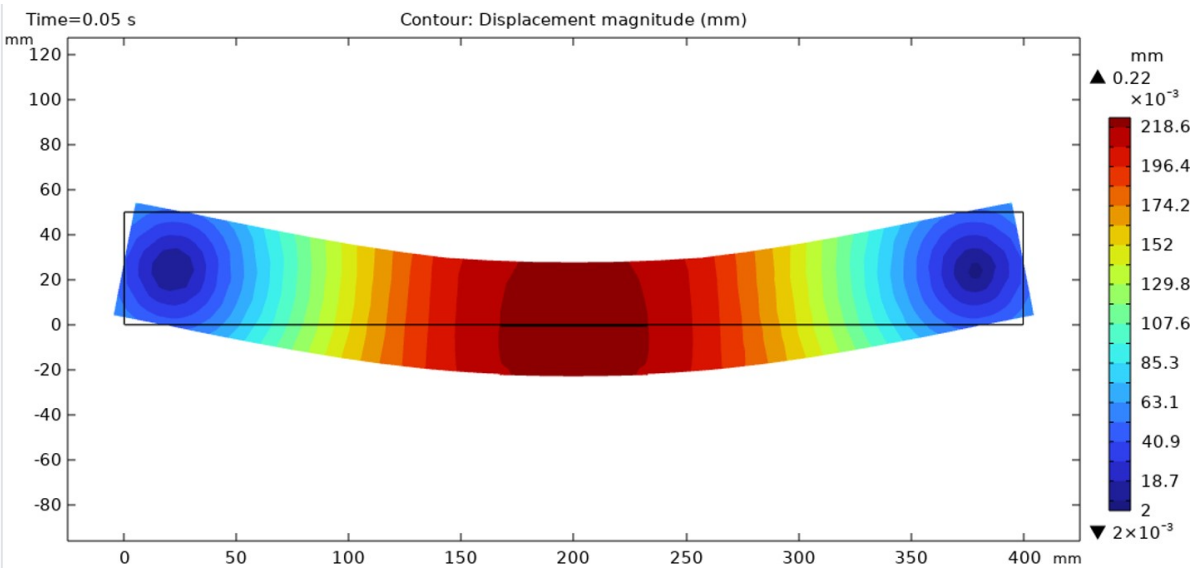


Figure G.6: COMSOL deflection for a strain of 400 $\frac{\mu m}{m}$