POLITECNICO DI TORINO

POLITECNICO DI TORINO

Master Degree course in Mechatronic Engineering

Master Degree Thesis

Evolution of a "Smart Product" project for a linear belt-driven motion system from TRL 4 to TRL 6.

Supervisors
Prof. Mihai Lazarescu
MSc Eng. Alfonso Picariello

Candidate
Hui Guo

Academic Year 2020-2024

Abstract

The objective of this work is to advance the development of intelligent products from Technology Readiness Level 4 to Technology Readiness Level 6. This advancement will be achieved by developing efficient data analysis methods and integrating algorithms with intelligent hinges, thereby enhancing the functionality and performance of these systems. This research is crucial as it addresses the growing need for precision and efficiency in various industries that rely on automation and intelligent technologies, particularly in the context of Industry 4.0 and smart edge solutions.The focus of this study is on linear belt drive motion systems and encoder detection systems. Linear belt drive systems are common in industrial processing and transportation, where the condition of the belt during transportation needs close attention to better monitor and prevent failures. Therefore, this work emphasizes developing algorithms to visually reflect the current state of the belt through the encoder detection system, integrating this with intelligent hinges to transition product testing from the laboratory to real-world environments.

Contents

List of Figures

Abbreviations

A-D anderson-darling(A-D) . . . 32

AI artificial intelligence(AI) . . . 11

CDF cumulative distribution function(CDF) . . . 33

DT decision tree(DT) . . . 16

EDF empirical distribution function(EDF) . . . 32

ELBO evidence lower bound(ELBO) . . . 46

FN flase negative(FN) . . . 50

FP false positive(FP) . . . 50

GUI graphical user interface(GUI) . . . 10

HSD honest significant difference(HSD) . . . 35

IIoT industrial internet of things(IIoT) . . . 12

IoT internet of things(IoT) . . . 10

K-S kolmogorov-smirnov(KS) . . . 32

KL kullback-leibler(KL) . . . 46

KNN k-nearest neighbors(KNN) . . . 49

LR logistic regression(LR) . . . 49

LSTM long short-term memory(LSTM) . . . 45

LSTM-VAE long short term memory and variational autoencoder (LSTM-VAE) . 47

MSE mean squared error(MSE) . . . 48

NB naive bayes(NB) . . . 49

PDF probability density function(PDF) . . . 32

Q-Q quantile–quantile(Q-Q) . . . 32

RMSE root mean square error(RMSE) . . . 48

RNN recurrent neural network(RNN) . . . 14

SVM support vector machine(SVM) . . . 14

TDF theoretical distribution function(TDF) . . . 32

Chapter 1

Introduction

As technology progresses, industries are increasingly adopting innovations that enhance efficiency and accuracy. Industry 4.0 is a pioneer of this transformation, characterized by the integration of digital technologies and intelligent systems into manufacturing and industrial processes. This shift necessitates the development of smart products that can adapt to dynamic environments and enhance operational performance.

A key component of Industry 4.0 is the deployment of intelligent edge solutions that utilize real-time data and advanced analytics to optimize industrial operations. In this context, the concept of technology readiness level(TRL) serves as a framework to assess the maturity of technologies, guiding their development from the conceptual stage to fully operational systems. This work focuses on advancing smart products from TRL 4 (validation of basic technological components in a laboratory environment) to TRL 6 (demonstration of these systems in relevant operational environments).

A specific area of focus in this research is the enhancement of linear belt-driven motion systems, which are indispensable in many industrial applications including manufacturing, material handling, and transportation. These systems require precise monitoring to prevent failures and ensure continuous high efficiency. The core of this research combines encoder detection systems with intelligent hinges, aiming to provide accurate real-time monitoring and control of belt conditions.

The problem addressed in this study is the need for reliable and efficient monitoring solutions for linear belt-driven systems. Current detection methods for belt systems often fail to provide the necessary precision and real-time data required for proactive maintenance and optimization. By developing complex algorithms that utilize encoder detection systems, this research strives to accurately reflect the operational status of belts, thereby preventing downtime and enhancing overall system performance. Ultimately, this not only saves production costs and labor but also significantly improves production quality and efficiency, fully aligning with the current smart manufacturing ideals of Industry 4.0.

Moreover, the entire process aims to bridge the gap between laboratory testing and actual application. By combining intelligent hinges with the developed monitoring algorithms, it ensures that these systems can be effectively tested and validated in environments simulating actual industrial conditions. This approach not only enhances the reliability and performance of the products but also ensures they are deployment-ready

for actual industrial scenarios.

In summary, this work contributes to advancing the TRL of smart products and the field of monitoring and maintaining belt systems, particularly focusing on linear belt-driven motion systems and encoder detection systems. It also helps to perfect the cloud-based GUI interface and enables the cloud to monitor the status of machines in real-time, laying the foundation for future analysis and product development.

The structure of this thesis is as follows:

- Chapter 2: This chapter discusses the integration of internet of things(IoT), AI, and edge computing within Industry 4.0, focusing on the application and maintenance of linear belt-driven motion systems, including the use of various algorithms and statistical methods for predictive maintenance.
- Chapter 3: This chapter provides an overview of TRL, emphasizing its importance in structured assessment, clear communication, and risk management during product development, particularly for conveyor belt systems integrated with IoT and edge computing.
- Chapter 4: This chapter details the use of encoder monitoring systems to detect belt tension and the methodology for collecting and analyzing data to monitor and maintain belt-driven systems, including statistical and algorithmic approaches to fault detection.
- Chapter 5:This chapter presents laboratory experiments for validating belt-driven systems at TRL 4-5, involving data processing, transformation methods, and statistical analysis to test and verify the system's reliability and performance.
- Chapter 6: This chapter describes experiments conducted in relevant environments to demonstrate belt-driven systems at TRL 5-6, focusing on data collection, analysis, and the application of machine learning models for fault detection and system validation.
- Chapter 7: This chapter focuses on incorporating the developed data processing algorithms into a comprehensive algorithm framework. It details the development of a graphical user interface(GUI) designed for the maintenance of running conditions.
- Chapter 8: This chapter summarizes the findings and achievements of the research, highlighting the advancements made in transitioning intelligent products from TRL 4 to TRL 6. It also outlines potential areas for future research and development, suggesting improvements and further applications of the developed systems and algorithms.

Chapter 2

Intelligent Technologies and Belt-Driven System Maintenance

The industrial landscape is significantly transformed by emerging technologies and Industry 4.0 principles. At the forefront, smart product development incorporates IoT, artificial intelligence(AI), edge and cloud computing, crucial for enhancing efficiency, product quality, and enabling predictive maintenance. Belt-driven systems, prevalent in industrial settings, necessitate regular upkeep due to their inherent characteristics. Hence, the adoption of intelligent detection and maintenance technologies is vital for their sustained reliability and longevity.

2.1 Background and Importance of Smart Product Development

The development of smart products has become a key focus in modern manufacturing and industrial sectors. The integration of advanced technologies such as the Internet of IoT, AI, and cloud computing is transforming traditional product development processes into intelligent, efficient, and highly adaptive systems.

The advent of Industry 4.0, often referred to as the Fourth Industrial Revolution, has had a significant impact on the development of smart products. This revolution is characterized by the integration of cyber-physical systems, IoT, and edge computing or cloud computing, collectively enhancing the connection and automation of industrial processes [7] as Figure 2.1. Industry 4.0 aims to create a highly flexible and efficient manufacturing environment where data from various sources, such as sensors and devices, is used to optimize production, improve quality, and reduce costs.

IoT technology is pivotal in smart factories, underpinning Industry 4.0 by facilitating real-time data exchange and communication across factory components. This technology transforms traditional manufacturing settings into highly efficient, adaptable systems [3]. Utilizing sensors and actuators, IoT creates a seamless network that monitors and controls production, ensuring factory-wide synchronized operations.

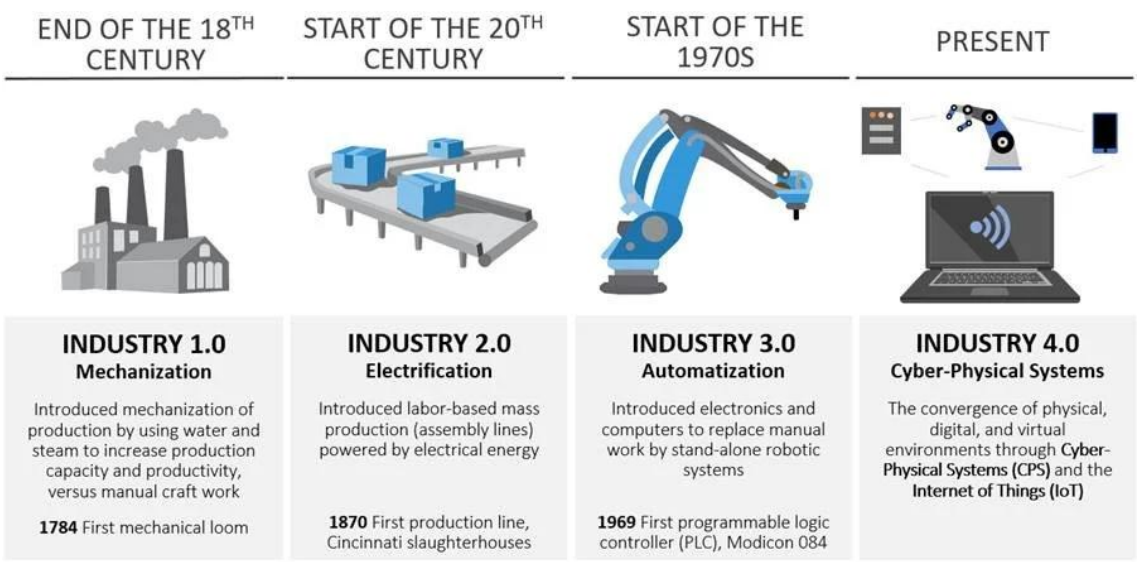

Figure 2.1: Industrial 4.0 Concept Diagram [7]

This connection not only enhances operational efficiency but also significantly improves the factory's responsiveness and adaptability to changes. The vast amount of data collected by IoT devices provides deep insights into the manufacturing processes, supporting data-driven decision-making that further optimizes production processes and resource utilization. This data-driven approach makes production flows more intelligent and precise, thereby increasing overall productivity and product quality.

Within the framework of Industry 4.0, IoT also facilitates the high degree of automation and self-optimization of production systems. Through horizontal and vertical data integration, IoT technology seamlessly connects suppliers, partners, and customers while achieving complete integration from product development to final production within the organization as Figure 2.2. This comprehensive integration enables production systems to respond quickly to market changes, meet personalized demands, and enhance production flexibility and customer value.

Edge computing plays a crucial role in the advancement of Industry 4.0 by addressing the limitations of traditional cloud computing in the Industrial industrial internet of things(IIoT). In Industry 4.0, edge computing enhances real-time data processing, reduces latency, and improves bandwidth efficiency by processing data closer to the data source. This decentralized approach ensures faster response times, critical for applications such as predictive maintenance, real-time quality control, and automated decision-making in manufacturing environments as Figure 2.3. For instance, edge computing allows for the immediate detection of anomalies in machine operations and enables rapid responses to potential issues, thereby minimizing downtime and optimizing production processes [13].

The integration of edge computing with IoT is fundamental to the realization of IIoT. Edge devices equipped with sufficient computing power can preprocess data locally, reducing the volume of data that needs to be sent to the cloud. This not only alleviates

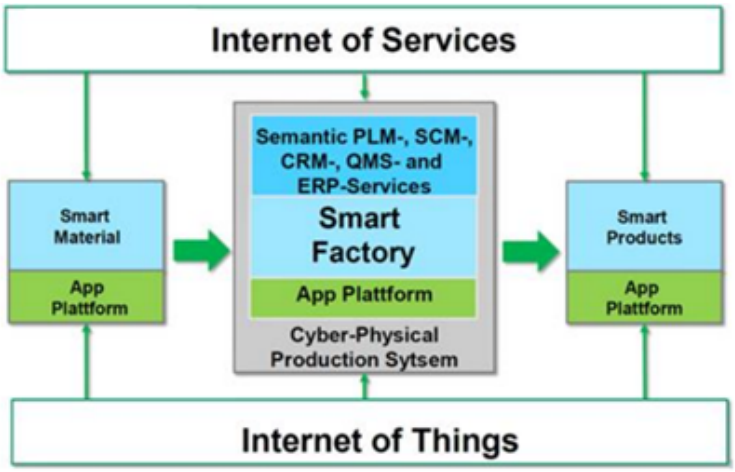

Figure 2.2: Smart Factory and IoT [3]

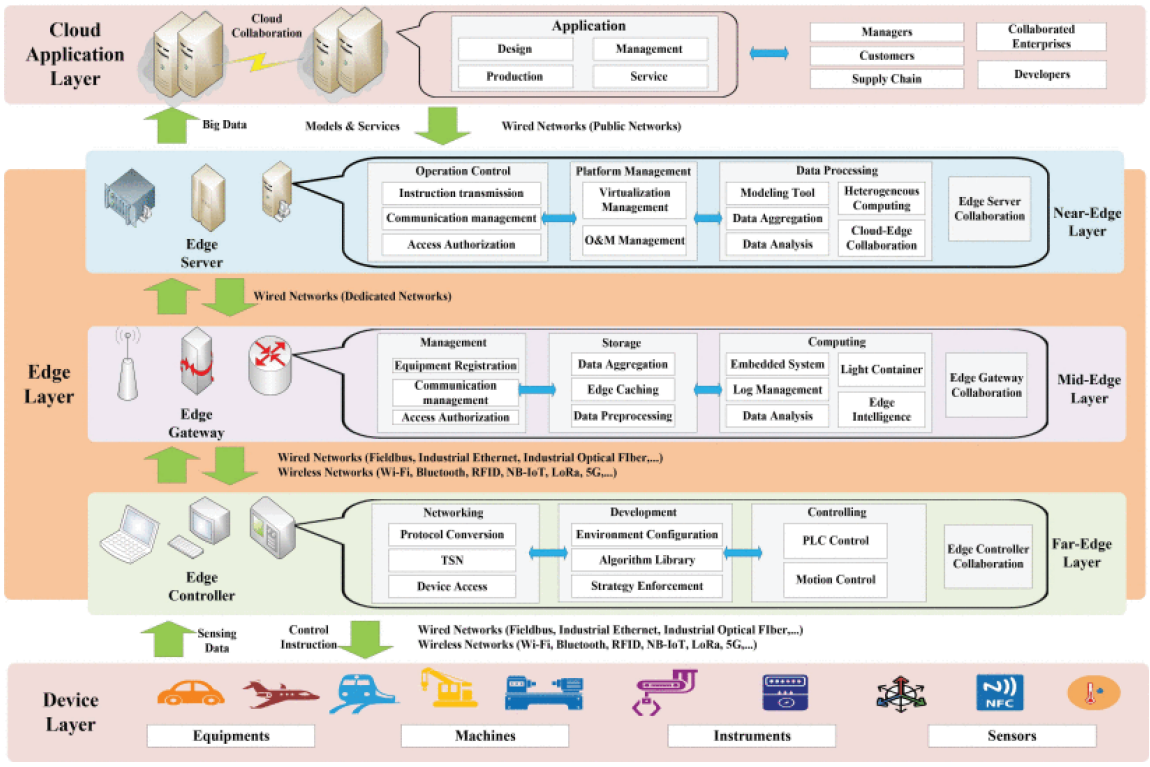

Figure 2.3: Reference Architecture of Edge Computing in IIoT [13]

network congestion but also enhances data security by minimizing the transfer of sensitive information over networks. In smart factories, for example, edge gateways collect and process data from various sensors and machines, enabling real-time monitoring and

control of production lines. This setup allows for more efficient and responsive manufacturing processes, contributing to the overall agility and productivity of Industry 4.0 initiatives [3].

Moreover, the synergy between edge computing and IoT extends to various industrial applications, where edge computing facilitates real-time monitoring and management of energy distribution, enhancing the reliability.

Edge computing significantly enhances the capabilities of Industry 4.0 by enabling real-time data processing, improving system performance, and ensuring data security [9]. Its integration with IoT forms the backbone of IIoT, driving innovations in various industrial sectors and paving the way for more responsive, efficient, and intelligent industrial operations, AI faces significant challenges related to the edge limitations of computing capacity within Industry 4.0. To effectively monitor and analyze shop floor machines using AI models, cloud services are necessary to address issues related to storage and computational resources. The pre-existing knowledge bases and big data stored in cloud servers can collaborate with edge computing, providing reliable and accurate information to edge devices [3].

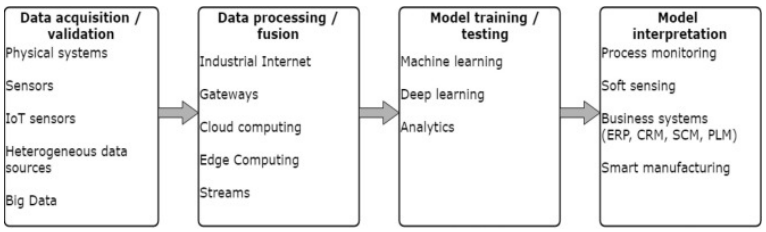

Figure 2.4: Industrial AI Data Pipeline [15]

In Industry 3.0, productive maintenance was optimized, and Total total productive maintenance(TPM) checklists were introduced, significantly reducing downtime and failure rates. However, achieving completely uninterrupted production was nearly impossible. In Industry 4.0, data-driven methods can determine machine status based on historical data sets and even predict potential downtimes. This predictive maintenance allows factories to plan alternative solutions and backups, laying the foundation for zero downtime.

Machine learning models are commonly used for predictive maintenance, where collected data can be utilized for maintenance and fault prediction [15] as Figure 2.4. Common maintenance models include support vector machine(SVM), Autoencoder, and recurrent neural network(RNN). These models enable the factory to anticipate issues and take preemptive actions, thus improving overall efficiency and reducing unexpected disruptions.

2.2 Overview of Linear Belt-Driven Motion Systems

Linear belt-driven motion systems are a crucial component in various automation and mechanical applications as Figure 2.5, providing precise linear movement with high speed and precision.

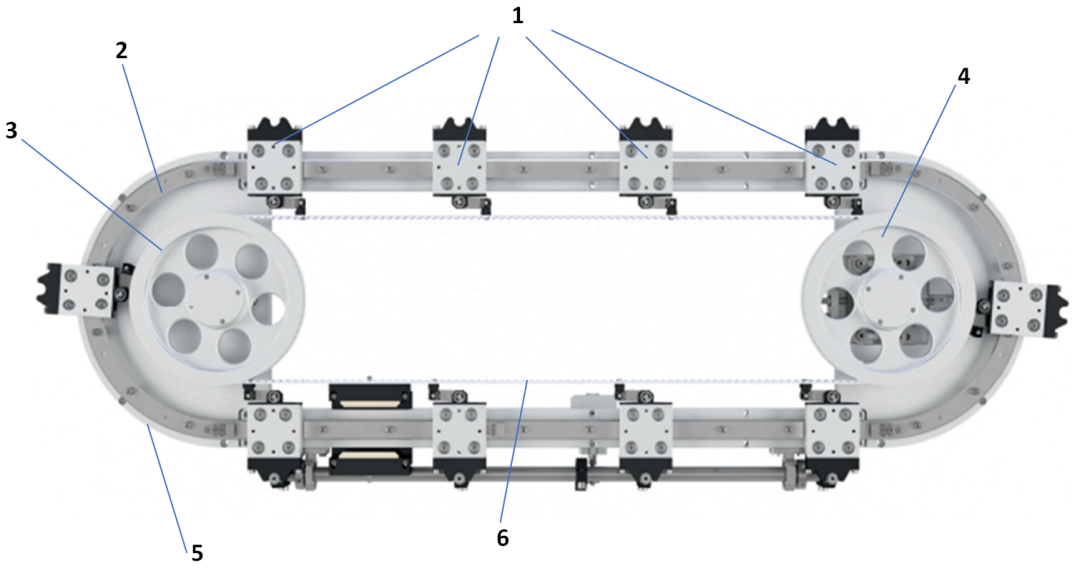

Figure 2.5: Structure of belt system

where 1. Carts 2. Circular Rail 3. Drive Pulley 4. Idler Pulley 5. Aluminum Base 6. Conveyor Belt Carriages are a critical component in the circular system, designed to support and guide loads along the circular rail. They are equipped with guide rollers that support the load in all directions. The standard carriages feature concentric and eccentric guide rollers which ensure precise movement on both the circular and straight sections of the rail.

The circular rail is composed of both circular and straight sections, induction hardened and ground to improve surface finish and accuracy. The rail's modular design allows for easy shipment and assembly, with alignment blocks simplifying the joining of rail sections.

The drive pulley is directly mounted on the gearbox output shaft to achieve maximum integration and dynamic performance. High-quality gearboxes with reinforced bearings support the belt tension and ensure precise movements.

The idler pulley in the system is mounted on a slide to allow for belt tensioning. This feature ensures that the toothed belt remains at the correct tension, providing reliable and smooth operation. Proper belt tensioning is crucial for maintaining the system's performance and longevity.

The system's base is constructed from machined aluminum alloy, providing a compact and space-saving solution. This base directly connects to the gearbox, ensuring stability and ease of mounting. The aluminum profiles supporting the linear guide rails have T-slots for fixing additional components like lubricators and carriage indexing cams, enhancing the system's flexibility and customization options.

The toothed conveyor belt is used for carriage traction in the system, offering a silent and maintenance-free transmission. The design minimizes the distance between the belt and carriage to reduce peak loads during high-speed transitions between straight and curved paths. The belt's length determines the module dimensions, and its tension is

maintained by the idler pulley, ensuring consistent and reliable operation.

Linear belt-driven motion systems are integral to modern automation solutions, offering a blend of precision, flexibility, and durability. Their application ranges from manufacturing lines to complex robotic systems, underlining their versatility and importance in various industrial contexts.

2.3 Current Maintenance Technologies in Linear Belt-Driven Systems

The reliability of linear belt-driven systems is a critical focus in the production process. Maintaining operational status and implementing predictive maintenance are essential to ensure system safety and reliability.

Jaroslaw and colleagues researched the maintenance of underground mine belt conveyors. They used robots to collect thermal imaging data of conveyor idlers to ascertain the current system status. Since the rolling resistance between the idlers and the moving belt affects belt wear, the thermal imaging data near the idlers serve as maintenance indicators, primarily using computer vision methods [8].

Xiangwei Liu and others detected the state of the belt pulley system by continuously measuring the temperatures of multiple idlers with different lifespans over several days. They simulated studies through constant inspection intervals. After some months of continuous operation, the normal distribution of the temperatures of more and more idlers shifted to a higher value range, indicating a significantly increased failure rate. This allows predicting the threshold for failures, helping to forecast the frequency and number of idler replacements within a certain period, thus aiding in overall system status detection and fault assessment [14].

D.Valis and colleagues employed a direct measurement method of belt extension since belt elongation directly affects system operation. They collected hundreds of data points on-site and established a Local Linear Model to assess the conveyor belt's status. Subsequently, they constructed Kalman filters, Kalman smoothers, and predictors to forecast the system's future state [4].

P.V.S.Anusha proposed a predictive maintenance system based on IoT, cloud storage services, and machine learning. This system successfully identified potential fault indicators, thereby reducing production downtime. The design primarily consists of hardware composed of sensors and ESP32 development boards, Amazon Web Services for data collection, storage, and analysis, and machine learning algorithms to develop conveyor belt fault prediction models. Multiple machine learning algorithms, such as decision tree(DT) and SVM, were compared [11].

Arthur Pollak and others conducted fault detection for belt drives based on the Industry 4.0 platform. They achieved online monitoring through sensor devices and cloud storage technology. Faults and normal equipment were then identified using the neural network Autoencoder algorithm, ultimately distinguishing between normal equipment data and assumed faulty equipment data [1].

Callum Webb and colleagues predicted the wear rate of mining conveyor belts. They

used cross-validation to evaluate the performance of the Random Forest and linear regression algorithm. Finally, they developed a predictive model applicable to other types of conveyors [2].

In this thesis, we adopt a simulation method for belt tension status, significantly reducing the development cycle. Statistical methods are used to estimate the trend of belt tension and predict fault occurrence. Lastly, combining IoT, smart hinge, and edge computing concepts, we design a product based on the Industry 4.0 concept.

2.4 Overview of the Methodological Approach

This section delves into the methodology used to advance the belt-driven system products from TRL 4 to TRL 6. The primary focus is on developing an effective and stable approach by combining technologies and processes encompassed within Industry 4.0 to monitor and maintain the operational status of this system. The key technologies utilized include IoT, edge devices, edge computing, cloud servers, statistical validation, and machine learning algorithm estimation as Figure 2.6.

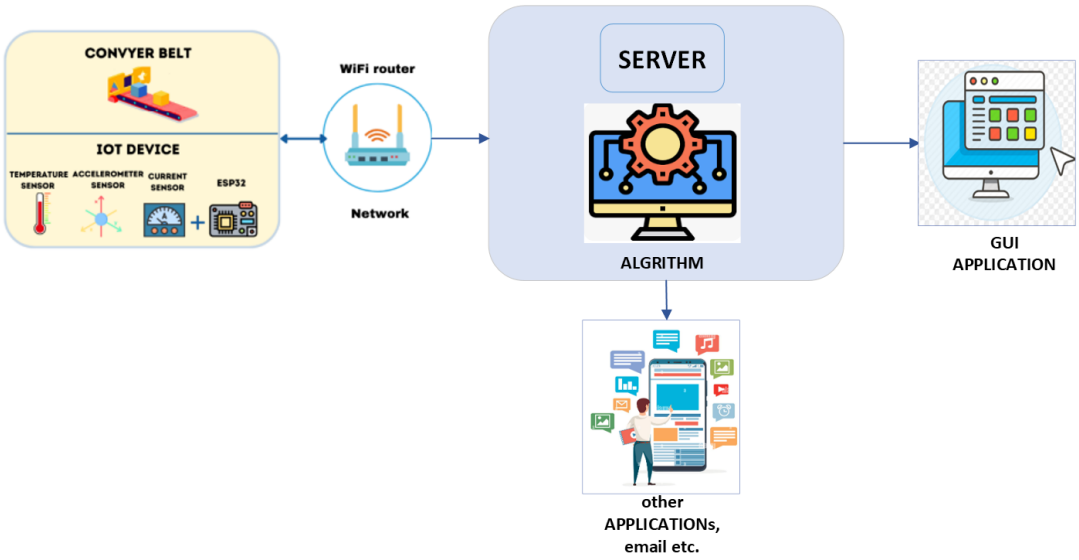

Figure 2.6: Methodological Approach of product maintenance

The first step involves using encoder devices to collect real-time data from the belt-driven system. Encoders are crucial for accurate and continuous feedback on the belt's status. This data is essential for identifying any deviations or anomalies that may indicate potential issues or inefficiencies within the system. During the transition from TRL 4 to TRL 6, data collection does not involve long-term continuous data collection from the conveyor system. Instead, the system's behavior under different stress conditions is assumed, which helps accelerate product development and ensure the reliability of the results.

Once data is collected, it is transmitted to edge computing devices located near the belt system, known as 'SMART EDGE'. These devices are capable of performing initial data processing and cleaning. The advantage of using edge devices is their ability to process data locally, reducing latency and enabling faster response times. Local processing includes filtering out noise and irrelevant data, compressing data to reduce transmission size, and extracting meaningful features critical for further analysis.

After initial processing at the edge, the data is sent via IoT technology to a central server. This server is part of the intelligent hinge platform, which integrates data sources from the system. At this stage, data undergoes more comprehensive analysis and is stored for historical tracking and deeper learning.

On the cloud server, advanced machine learning and deep learning algorithms are applied to the data. These algorithms aim to predict potential faults and maintenance needs by learning from historical data patterns and current operational indicators.

A GUI on the server allows for real-time monitoring and visualization of the system status. This interface provides operators and maintenance teams with a user-friendly way to view and interpret data. It displays various metrics, such as current operational status, trend analysis, and alerts for any detected anomalies.

Additionally, the system incorporates communication tools like email and instant messaging to automatically notify maintenance personnel and system operators of potential issues. This proactive communication helps schedule maintenance activities in a timely manner and prevents unexpected downtimes.

The cloud platform continuously updates predictive models based on new data and insights gained from ongoing operations. This iterative learning process ensures the system evolves and adapts to changing conditions in the operational environment, thereby enhancing the reliability and efficiency of the predictive maintenance system.

In summary, advancing the belt-driven system from TRL 4 to TRL 6 involves a combination of real-time data collection, edge computing, IoT integration, and advanced data analysis hosted on a cloud platform. This integrated approach ensures the system is not only effectively monitored and maintained but also capable of adapting to new challenges and demands in industrial environments.

Chapter 3

Definition and Significance of Technology Readiness Level

TRL serve as a crucial framework for assessing and communicating the maturity of a technology. Originally developed by NASA, TRL has been widely adopted across various industries including defense, aerospace, energy, and manufacturing, underscoring its versatility and significance in guiding technology development. The structured assessment provided by TRL aids in identifying and mitigating risks early in the development process, facilitates clear and consistent communication among stakeholders, ensures effective resource allocation, and enhances risk management. This chapter also explores the application of TRL in developing conveyor belt systems integrated with Industry 4.0, IoT, and edge computing technologies, and outlines the implementation methods for advancing these systems from TRL 4 to TRL 6 through systematic validation and integration. [5]

3.1 Overview of Technology Readiness Levels

TRL is an important framework for assessing and communicating the maturity of a technology. TRL scale was originally developed by NASA in the 1970s to evaluate the readiness of technologies for space missions and has since been adopted by various industries [5], including defense, aerospace, energy, and manufacturing. This adoption highlights its versatility and significance in guiding technology development. The levels are as Figure 3.1.

The use of TRL is particularly important in product development and research for the following reasons:

Structured Assessment: TRL provides a structured and standardized method to assess the progress of technology development from basic research to full deployment. This structure helps identify and mitigate risks early in the development process.

Clear Communication: By providing a common language and set of standards, TRL facilitates clear and consistent communication among stakeholders, including developers, investors, and regulatory bodies. This clarity helps set realistic expectations and timelines.

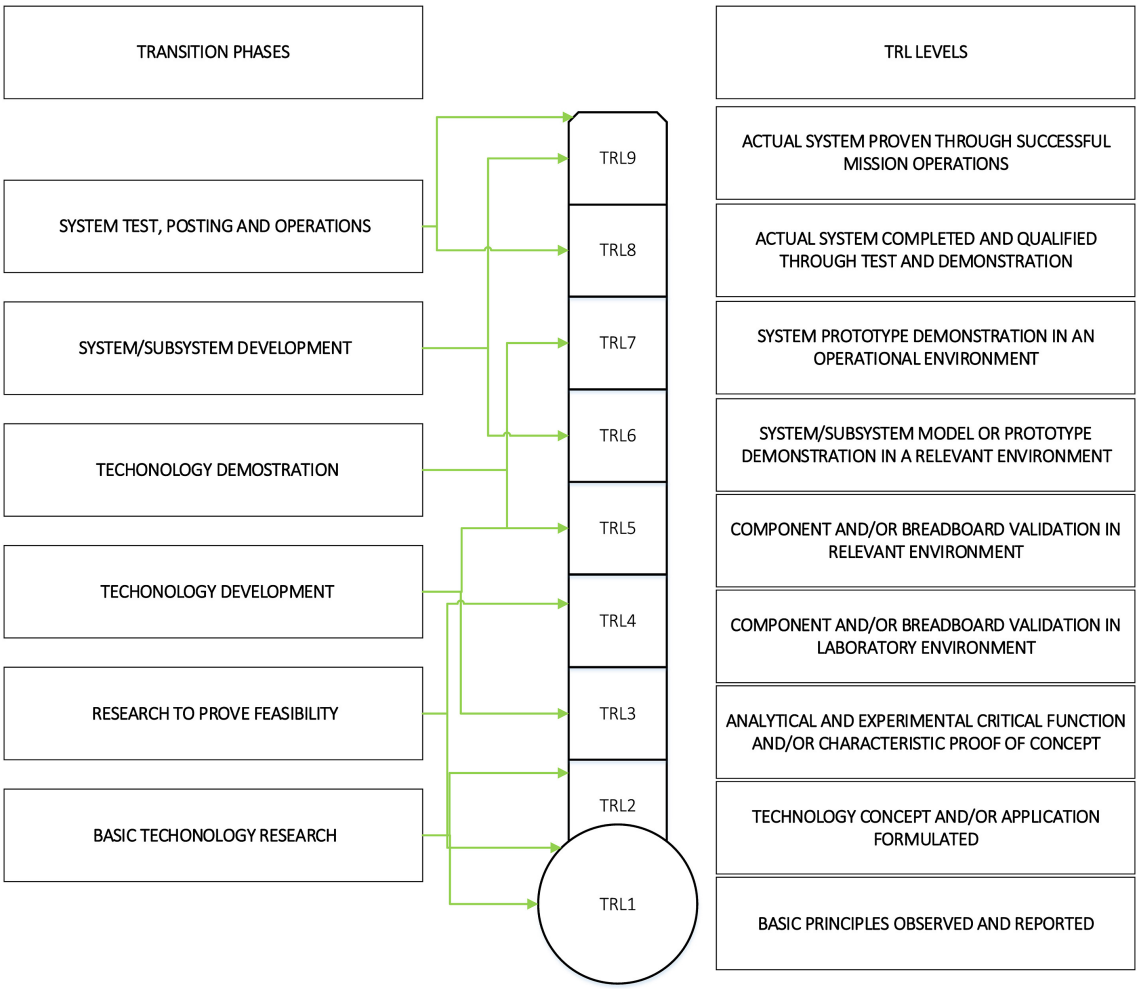

Figure 3.1: Transition Phases and Technology Readiness Levels (TRL) [5]

Resource Allocation: The systematic approach of TRL helps in the effective allocation of resources by emphasizing the readiness level of a technology. This enables managers to make informed decisions about where to focus investments and efforts.

Risk Management: TRL helps identify potential risks at various stages of technology development. Understanding these risks early allows developers to implement appropriate mitigation strategies, increasing the likelihood of successful project completion.

Advantages of TRL in Developing Industrial 4.0, IoT, and Edge Computing Products Enhanced Decision-Making:Provides a clear framework for assessing technological progress, aiding in making informed decisions about continuing, modifying, or halting projects [12].

Improved Efficiency: Ensures the efficient use of resources by focusing on technologies that are ready to progress to the next stage of development.

Increased Stakeholder Confidence:Demonstrating a systematic approach to technology

development can enhance stakeholder confidence and attract potential investors.

Application of TRL in Conveyor Belt Systems for Industry 4.0, IoT, and Edge Computing. In the context of developing conveyor belt systems integrated with Industry 4.0, IoT, and edge computing technologies, transitioning from TRL 4 to TRL 6 is crucial.

TRL 4 to TRL 5: At this stage, the system is validated in a laboratory environment. For instance, IoT sensors and edge computing modules can be tested under controlled conditions for data collection, processing, and transmission.

TRL 5 to TRL 6: This involves demonstrating the integrated system in a relevant environment. Here, the system with IoT and edge computing technologies can be deployed in a pilot industrial environment to verify its performance, reliability, and operational benefits.

Using the TRL framework for this development ensures systematic validation of each technological component and their integration, reducing the risk of failures during full-scale implementation. It also helps address potential issues related to interoperability, scalability, and real-time data processing, which are critical for the success of Industry 4.0 applications.

In conclusion, the TRL framework is an indispensable tool for guiding the development of innovative technologies, ensuring they reach the necessary maturity levels before large-scale deployment. Its application in developing advanced conveyor belt systems for Industry 4.0, IoT, and edge computing will help achieve reliable, efficient, and scalable solutions.

3.2 Implementation Methods for Advancing From Technology Readiness Level 4 to 6

The laboratory prototype and relevant environment prototype of the Linear Belt-Driven Motion Systems product have both been designed and are ready for experimentation. The two prototypes differ in size; the laboratory prototype is intended to test functionality and the effectiveness of algorithms, while the relevant environment prototype is used to validate the applicability of the functionality and algorithms.

Transition from TRL 4 to TRL 5: Initially, data collected from the prototypes is cleaned. The next step is to preprocess the data. If any errors are found during the preprocessing stage, the data will need to be collected again until a suitable and stable collection method is found. Data analysis is conducted according to the laboratory environment settings, continuously seeking suitable analytical methods to achieve results that meet the initial expectations. The distinction between normal and fault states is made, and statistical tests are performed on different data distributions. Fault thresholds are defined, and future states are predicted. During this stage, the analytical methods developed are not deployed to the edge devices nor integrated with the algorithms of the edge devices, as this stage is solely for laboratory testing. The matured algorithms will be deployed to edge devices in the next stage.

Transition from TRL 5 to TRL 6: In the second stage, data is collected using the same methods and processed and analyzed using the laboratory methods on data from the relevant environment. The obtained results are compared with the expected outcomes.

If the results do not meet expectations, the processing and analysis methods are adjusted to fit the current environment's data. Similar to the previous stage, fault thresholds and methods to distinguish fault states, along with related statistical results, are determined. Usable algorithms are then integrated with the pre-deployed algorithms on the edge devices. Finally, these are deployed to the edge devices and connected to the cloud server. A GUI interface is developed to display and monitor the product's status and all available information.

This structured approach as the Figure 3.2, ensures that each stage of the development process is thoroughly tested and validated, leading to a reliable and efficient final product ready for deployment in real-world industrial settings.

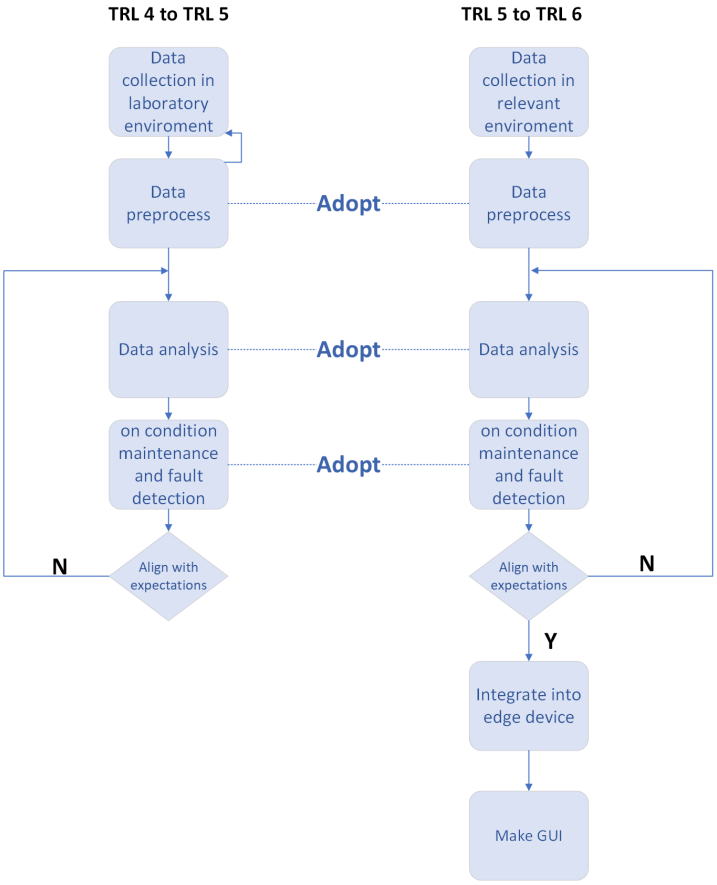

Figure 3.2: Process Flow from TRL 4 to TRL 6

Chapter 4

The Data Collection for the Linear Belt-Driven Motion System

4.1 Introduction to Encoder Monitoring Systems

The encoder detection system is integral to monitoring the operational integrity and longevity of belt-driven mechanisms. The primary function of this system is to detect variances in belt tension that can indicate belt wear or potential failure. This detection is crucial for preventative maintenance and ensuring operational efficiency.

The encoder collects information about the belt, focusing primarily on two types of pulleys: the motor-driven pulley and the passive idle pulley. Both pulleys have holes in their structures. The encoder measures the time it takes for these holes to pass by, which allows it to determine the time difference between the passive and active pulleys. This time difference is then correlated with the tension of the belt.

The variation in belt tension can be considered inversely proportional to the time difference. As the tension decreases, it becomes more difficult for the drive pulley to move the driven pulley, resulting in a greater time difference. Consequently, greater tension corresponds to a smaller time difference detected by the encoder. Therefore, the time difference variable is crucial for state estimation and fault detection.

The information collected by the encoder can be divided into two main conditons due to the uncertainty of the initial states of the two pulleys. The position of the holes determines which pulley's measurement signal is detected first. Therefore, the two main conditons are as follows.

The transition signal from the motor-driven pulley is detected before the transition signal from the passive idle pulley, and the transition signal from the passive idle pulley is detected before the transition signal from the motor driven pulley.

These two conditons arise because the starting positions of the pulleys are not fixed. Depending on the initial alignment of the holes on the pulleys, the encoder will detect the transition of one pulley before the other. This initial detection sequence affects how the subsequent timing differences are interpreted and consequently how the belt tension

is evaluated.

The detailed analysis of these conditons allows for accurate monitoring of belt tension by correlating the timing differences between the pulleys' transitions. This ensures that variations in belt elasticity can be detected early, facilitating preventative maintenance and reducing the risk of unexpected belt failures.

The encoder collects information under the case that the first high-to-low transition of the motor-driven pulley's signal occurs before that of the passive idle pulley. Any signal from the initial condition to on-condition. The special sign "DC" means "don't care",in other words,we can neglect the initial condition of any signal because it is not related to our anylysis. Under this case, there are four classic case:

- Both the motor-driven pulley's and the passive idle pulley's encoder signals transition from low to high.

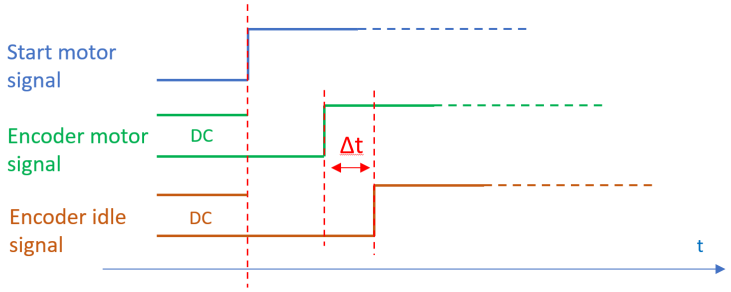

Figure 4.1: Case 1: both the motor encoder signal and the idle encoder signal present a transition from Low to High.

- Both the motor-driven pulley's and the passive idle pulley's encoder signals transition from high to low.

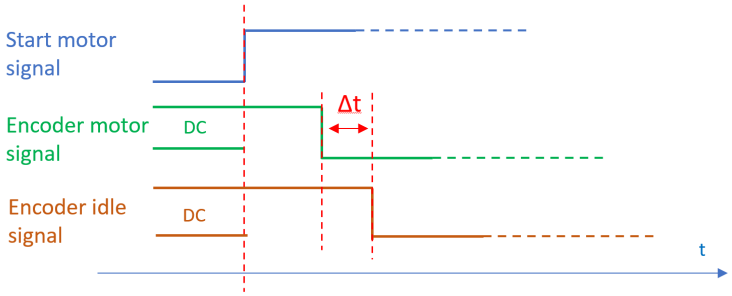

Figure 4.2: Case 2: both the motor encoder signal and the idle encoder signal present a transition from High to Low.

- The motor-driven pulley encoder signal transitions from low to high, while the passive idle pulley encoder signal transitions from high to low.

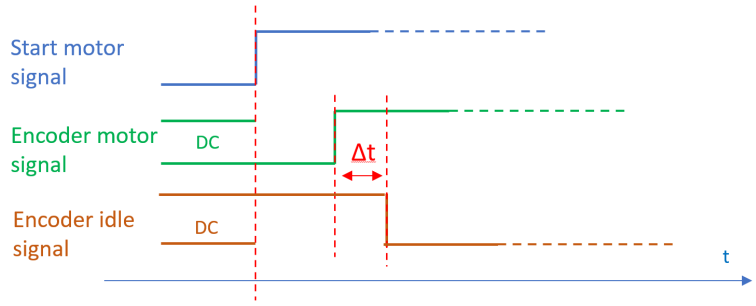

Figure 4.3: Case 3: the motor encoder signal presents a transition from Low to High, while the idle encoder transitions from High to Low.

- The motor driven pulley encoder signal transitions from high to low, while the passive idle pulley encoder signal transitions from low to high.

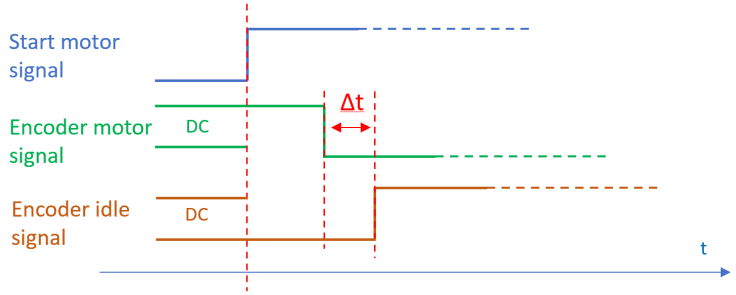

Figure 4.4: Case 4: the motor encoder signal has a transition from High to Low while the idle encoder from Low to High.

Another special case arises in which the first transition of the idler pulley encoder occurs before the first transition of the motor encoder pulley. In this case, there are also four sub-cases, which are as follows:

- The motor encoder signal transitions from Low to High while the first idle encoder signal transitions from High to Low then to high. Initially, the idle pulley's signal is high, and the motor-driven pulley's signal is low. This indicates that the motor-driven pulley begins to detect activity while the idle pulley reduces its detected activity.

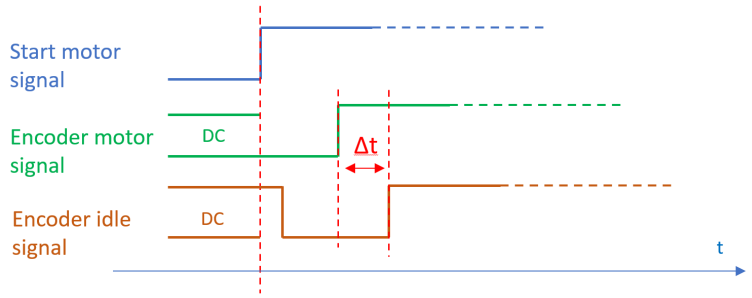

Figure 4.5: Case 1: The motor encoder signal has a transition from Low to High while the first idle encoder transitions from High to Low.

- The motor encoder signal transitions from High to Low while the first idle encoder signal transitions from Low to High then to low. Initially, the motor-driven pulley's signal is high, and the idle pulley's signal is low. This suggests that the motor-driven pulley reduces its activity while the idle pulley begins to detect increased activity.

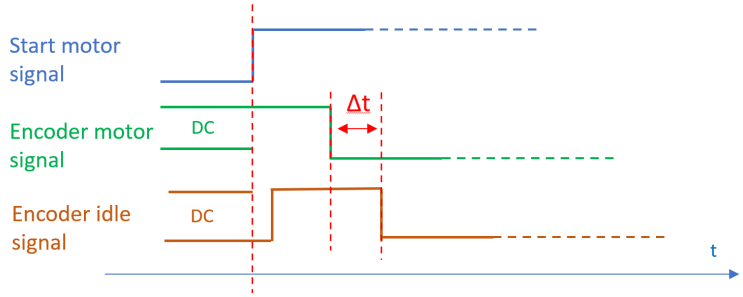

Figure 4.6: Case 2: The motor encoder signal has a transition from High to Low while the first idle encoder transition from Low to High.

- Both the motor encoder signal and the idle encoder signal have an initial transition from Low to High then idle encoder will be to low. In this case, both pulleys initially have low signals. This simultaneous increase in activity could indicate synchronized changes in the belt's tension or motion.

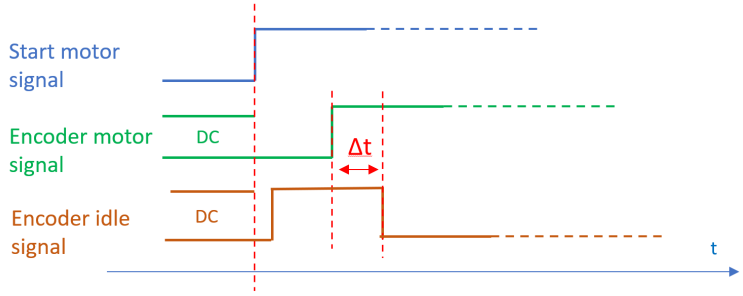

Figure 4.7: Case 3: both the engine and neutral encoder signals have an initial transition from Low to High.

- Both the motor encoder signal and the idle encoder signal have an initial transition

from High to Low then idle encoder will be to high. Initially, both pulleys have high signals. This simultaneous decrease in activity suggests a uniform change in the belt's condition or movement.

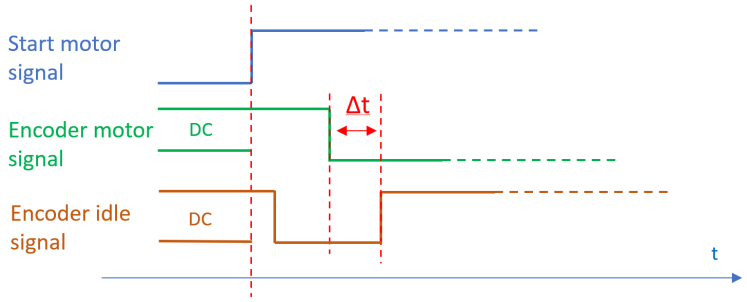

Figure 4.8: Case 4: both the engine and neutral encoder signals have an initial transition from High to Low.

By analyzing signal transitions detected by the encoder, the system can determine the timing differences between pulley signals, thereby providing insights into belt tension. This differentiation is crucial for accurately assessing the operational status and ensuring the proper functioning of belt-driven mechanisms. To effectively analyze these transitions, the data must be divided into eight distinct groups during data processing. This classification enables the distinction between various cases under which the data was collected. By comparing the statistical performance of different datasets under identical cases, the impact of different belt tensions can be more clearly illustrated. Consequently, this approach facilitates a detailed analysis of belt usage and opera.

4.2 Analysis of Belt Inspection Algorithms

In the previous section, we discussed the relationship between belt tension and time difference. We understand that the belt tension decreases over time during the system's operation until the belt's tensioning function fails. Therefore, it is crucial to halt the system before the belt tension reaches a critical state.

Given the robust and stable design of the linear belt-driven motion system and the good working environment of the belt system with minimal external noise, we assume that the time difference corresponding to the belt tension will exhibit a linear trend over extended periods of operation. This assumption eliminates the need for long-term observation and data collection of the belt system. Instead, we can simulate multiple similar belt tension states to test the algorithm's effectiveness.

The belt tension can be manually adjusted through the mechanical structure of the conveyor system, allowing the simulation of different tension levels and their corresponding encoder data distributions. By analyzing these data distributions, we can derive one or several statistical metrics to describe the belt's performance under current cases. Based on our assumption, the belt tension should exhibit a trend similar to that illustrated in Figure 4.9.

By using these simulated tension states, we can efficiently test and validate the effectiveness of our inspection algorithms, ensuring that the belt system operates reliably and

safely without the need for extensive long-term data collection. This approach allows for timely intervention before the belt reaches a critical tension state, thus maintaining the system's integrity and performance.

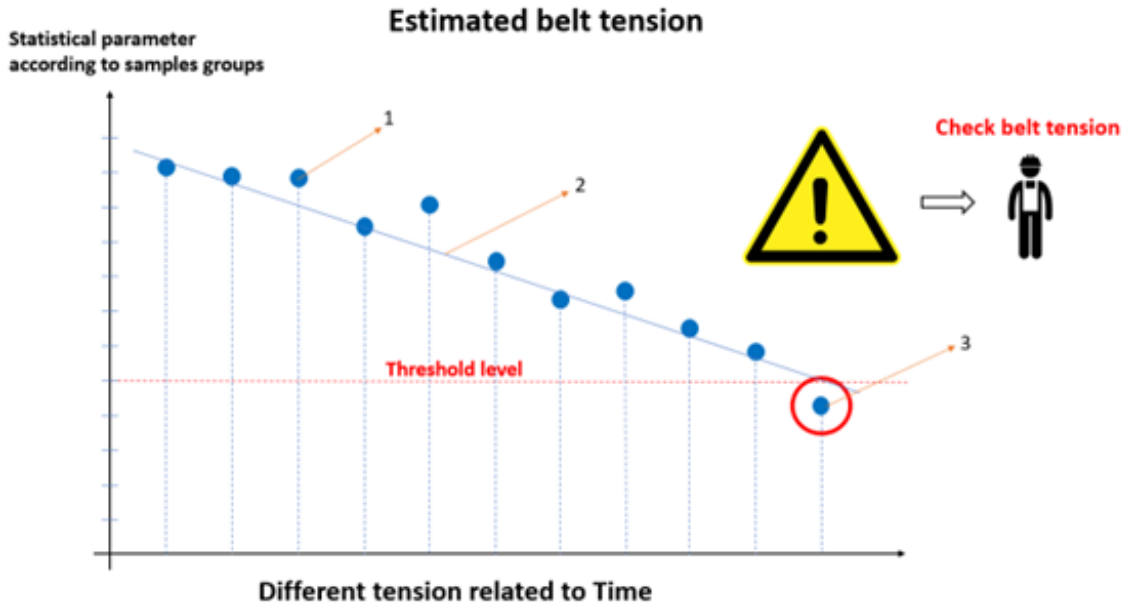

Figure 4.9: Estimated Belt Tension Over Time

In Figure 4.9, where 1.Statistical Descriptors of Time Difference Data Distribution for Different Belt Tensions 2.Estimated Trend of Belt Tension over Extended Operation 3.Estimated Fault States

As shown in the figure, this experiment simulates continuous tension changes as discrete tension changes. This approach significantly reduces product development time and increases the stability of the developed algorithms. The linear assumption has been proven effective with small prototypes in the early TRL stages.

This study aims to conduct state detection and fault analysis on a larger system. If data distributions under different belt tensions show differences and trends, future belt states can be predicted.

First, the encoder will distinguish eight different conditons for data analysis. The analysis involves distinguishing these eight conditons and performing the corresponding data analysis. Pseudocode shown as below:

This pseudocode initializes by reading the initial states of the motor encoder and the idle encoder. Then, it determines whether the event sequence belongs to a "typical" or "special" case based on the first transition of the motor encoder relative to the idle encoder. For each encoder sample, the pseudocode checks for transitions, records the time difference (ΔT) based on the specific situation, and finally calculates the weighted ΔT according to the sequence and time of transitions.

The resulting ΔT data distribution for each situation will be either quasi-Gaussian

Algorithm 1: Analysis of Encoder Transition Timings

Data: Initial samples for motor encoder and idle encoder
Result: Analysis based on encoder transitions and timing
Begin
Initialize list ΔT
Read initial samples for motor encoder and idle encoder
if *first transition of motor encoder occurs before idle encoder* **then**
  Set $flag = 0$ (Classic Case)
else
  Set $flag = 1$ (Special Case)
end
foreach *sample in motor and idle encoders* **do**
  Check transition type for motor encoder
  Check transition type for idle encoder
  if *motor encoder transitions from Low to High* ***and*** *idle encoder from High to Low* **then**
    Append ΔT with current timestamp difference (Case 1 or Case 5)
  else if *motor encoder transitions from High to Low* ***and*** *idle encoder from Low to High* **then**
    Append ΔT with current timestamp difference (Case 2 or Case 6)
  else if *motor encoder transitions from Low to High* ***and*** *idle encoder transitions from Low to High* **then**
    Append ΔT with current timestamp difference (Case 3 or Case 7)
  else if *motor encoder transitions from High to Low* ***and*** *idle encoder transitions from High to Low* **then**
    Append ΔT with current timestamp difference (Case 4 or Case 8)
end
if *flag == 0* **then**
  Classical case analysis complete;
else
  Special case analysis complete;
end
Calculate weighted ΔT based on the sequence and timing of transitions
End

or quasi-Poisson. If the data is not quasi-Gaussian, statistical data transformation methods are employed to convert the data distribution into a quasi-Gaussian distribution, facilitating subsequent analysis.

Once a quasi-Gaussian distribution is obtained, statistical tests are conducted on the data to analyze the distribution corresponding to each different tension, yielding differential data. The magnitude of these differences is then compared to distinguish the system's state. Since tension is adjustable, the analysis is conducted under known tension cases. Subsequently, the trend is estimated based on the statistical measures of all data. After obtaining the trend, an algorithm or machine learning model is used to detect faults.

Chapter 5

Experiments for Technology Readiness Level 4 to 5 Validation and Result

5.1 Data Processing in a Laboratory Environment

In a laboratory environment, time difference ΔT data corresponding to different belt tensions of the Linear Belt-Driven Motion Systems is collected. First, all time difference data is extracted from the raw data, and non-positive invalid values are removed. The data is then divided into eight different categories based on the varying cases of encoder data collection. Two approaches are attempted from this point: one involves analyzing the original quasi-Poisson distribution of the data, and the other involves performing data transformation on the dataset and then analyzing the quasi-Gaussian distribution data. Since the Poisson distribution does not require further processing, only the second approach is discussed here.

To make the data distribution conform to a quasi-Gaussian distribution, data transformation methods are used. Effective data transformation methods for the existing data include logarithmic transformation, Box-Cox transformation, Yeo-Johnson transformation, etc. At this stage, the effects of various feasible data transformation methods are compared, and one or several effective methods are selected as the processing approach.

Using 690 N as the dataset for verifying the data transformation method, the probability distribution plots of the originally classified data are drawn. The original data is observed to exhibit a right-skewed distribution characteristic. Effective processing methods for right-skewed data include Box-Cox transformation, Yeo-Johnson transformation, Cube transformation, Log transformation and Arcsin transformation.

The Box-Cox transformation is used to stabilize variance and make the data more normally distributed. The formula is:

$$y_i(\lambda) = \begin{cases} \frac{y_i^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0 \\ \ln(y_i) & \text{if } \lambda = 0 \end{cases} \tag{5.1}$$

where y_i is the original data, and λ is the transformation parameter. Yeo-Johnson transformation extends Box-Cox transformation to allow for zero and negative values. The formula is:

$$y_i(\lambda) = \begin{cases} \frac{(y_i+1)^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0, y_i \geq 0 \\ \ln(y_i + 1) & \text{if } \lambda = 0, y_i \geq 0 \\ -\frac{(-(y_i+1))^{2-\lambda} - 1}{2-\lambda} & \text{if } \lambda \neq 2, y_i < 0 \\ -\ln(-(y_i + 1)) & \text{if } \lambda = 2, y_i < 0 \end{cases} \tag{5.2}$$

The cube transformation raises each data point to the third power. The formula is:

$$y_i' = y_i^3 \tag{5.3}$$

where y_i is the original data, and y_i' is the transformed data. The log transformation is used to reduce skewness. The formula is:

$$y_i' = \ln(y_i) \tag{5.4}$$

where y_i is the original data, and y_i' is the transformed data. Note that for non-positive data, you need to add a constant to make all data positive before applying the transformation.

The Arcsin transformation is commonly used for data that are proportions or percentages. The formula is:

$$y_i' = \arcsin(\sqrt{y_i}) \tag{5.5}$$

where y_i is the original data (usually in the range [0, 1]), and y_i' is the transformed data. All the results of different data transform methods can be observed according to probability density function(PDF) plot and quantile–quantile(Q-Q) plot, as Figure 5.1 and Figure 5.2.

According to the PDF, it can be seen that the five methods are to some extent similar to the Gaussian distribution. Through the Q-Q plots, it is observed that the box-cox, Yeo-Johnson, log, and cube methods are closer to the assumed normal distribution.

The kolmogorov-smirnov(KS) test and anderson-darling(A-D) test are used for statistical testing to determine whether the distribution of the data, after being processed by various data transformation methods, is closer to a normal distribution.

The K-S test is a non-parametric statistical method used to compare the differences between a sample distribution and a reference probability distribution, or to compare the differences between two sample distributions. In this context, the K-S test is employed to determine whether the dataset conforms to a normal distribution.

First, calculate the empirical distribution function(EDF). Given a sample dataset, the empirical distribution function is defined as:

$$F_n(x) = \frac{1}{n} \sum_{i=1}^{n} 1_{\{x_i \leq x\}} \tag{5.6}$$

where **1** is the indicator function, which equals **1** when x_i is less than x, and 0 otherwise. Next, calculate the theoretical distribution function(TDF).

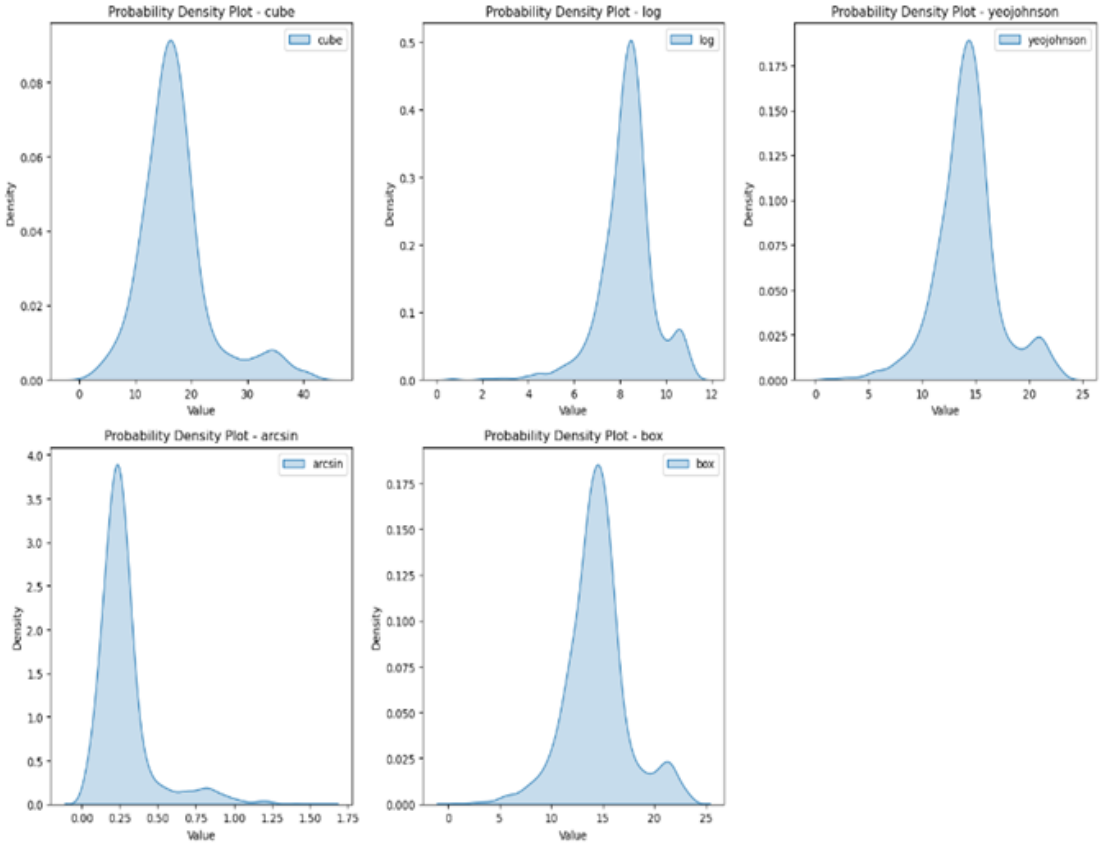

Figure 5.1: PDF plot for 5 different data transformation methods

For a normal distribution, the theoretical distribution function is the cumulative distribution function(CDF) of the standard normal distribution:

$$F(x) = \Phi(x) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right] \tag{5.7}$$

where μ and σ are the sample mean and standard deviation, respectively, and erf is the error function.

The K-S statistic value D is defined as the maximum difference between the empirical distribution function and the theoretical distribution function.

Finally, calculate the p-value and make a decision. Based on the sample size n and the K-S statistic D, use computational tools to determine the p-value. If the p-value is less than the significance level of 0.05, the null hypothesis is rejected, indicating that the data does not follow a normal distribution.

The A-D test is a statistical method used to assess whether a sample data set follows a specific distribution. Similar to the K-S test, the A-D test is also a non-parametric test but is more sensitive to the tails of the distribution. The A-D test is particularly suitable for testing whether data follows a normal distribution.

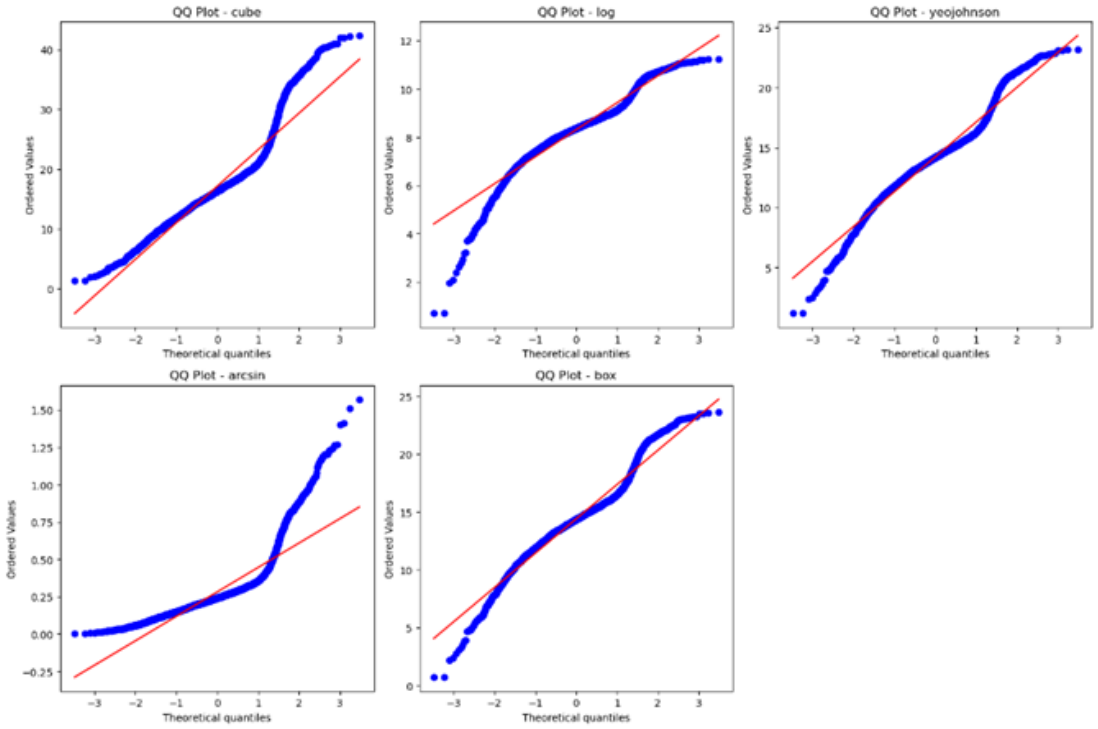

Figure 5.2: Q-Q plot for 5 different data transformation methods

First, sort the sample data in ascending order, then calculate the CDF using the same formula as the K-S test. Next, compute the A-D statistic as follows:

$$A^2 = -n - \frac{1}{n}\sum_{i=1}^{n}\left[(2i-1)\ln(F(x_{(i)})) + (2(n-i)+1)\ln(1-F(x_{(n-i+1)}))\right] \tag{5.8}$$

where n is the sample size, x_i is the i-th ordered data point, and F(x) is the theoretical distribution function.

Based on the A-D statistic, use computational tools to determine the p-value. If the p-value is less than the significance level, reject the null hypothesis, indicating that the data does not follow a normal distribution.

Table 5.1: **Comparison of Transformation Methods**

Transformation	**K-S P-value**	**A-D Statistic**
CUBE	8.77×10^{-41}	78.00
LOG	8.58×10^{-18}	41.00
YEO-JOHNSON	1.42×10^{-20}	37.64
ARCSIN	4.71×10^{-96}	192.00
BOXCOX	9.42×10^{-21}	37.92

According to table 5.1, Although the K-S test P-values for LOG , Yeo-Johnson Transformation are both very small, they are relatively larger among all the transformation methods, which suggests that their deviation from the normal distribution may be relatively smaller.The comparison of the A-D statistic to its critical value for Box-cox Transformation, along with a relatively higher K-S P-value, also indicates that it may be closer to a normal distribution, although the difference remains significant. Arcsin Transformation showed the greatest deviation from normal distribution, with both K-S and A-D tests clearly indicating a significant difference from the normal distribution.

Conclusion of comparison:

Based on the K-S P-values and A-D statistics, LOG Transformation and Yeo-Johnson Transformation exhibit relatively smaller deviations from normal distribution. Box-cox Transformation also shows better results, but slightly less so. CUBE Transformation and Arcsin Transformation significantly deviate from the normal distribution, especially Arcsin Transformation, which shows the greatest deviation in both tests.

Therefore, if we must choose the transformation that best conforms to the Gaussian distribution, Yeo-Johnson Transformation might be the best choice, closely followed by LOG Transformation. These two transformations provide the data distributions that are relatively closest to the normal distribution, although it is still important to note that all transformed data did not fully meet the strict definition of a Gaussian distribution.

5.2 Data Analysis in a Laboratory Environment

After processing the data, both the transformed and untransformed datasets were analyzed. Initially, data distribution plots and box plots were created to compare the relationships between the datasets. Subsequently, ANOVA and Tukey honest significant difference(HSD) tests were employed to quantify and compare the differences in the results.

To compare the differences among the three datasets, a one-way ANOVA was conducted. The ANOVA test uses the following formula to calculate the F-statistic:

$$F = \frac{MS_{\text{between}}}{MS_{\text{within}}} \tag{5.9}$$

Where MS_{between} (Mean Square Between Groups) is calculated as:

$$MS_{\text{between}} = \frac{SS_{\text{between}}}{df_{\text{between}}} \tag{5.10}$$

SS_{between} (Sum of Squares Between Groups) measures the variance between the means of the groups:

$$SS_{\text{between}} = \sum_{i=1}^{k} n_i (\bar{X}_i - \bar{X}_{\text{grand}})^2 \tag{5.11}$$

SS_{within} (Sum of Squares Within Groups) measures the variance within each group:

$$SS_{\text{within}} = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (X_{ij} - \bar{X}_i)^2 \tag{5.12}$$

df_{between} (Degrees of Freedom Between Groups) is:

$$df_{\text{between}} = k - 1 \tag{5.13}$$

df_{within} (Degrees of Freedom Within Groups) is:

$$df_{\text{within}} = N - k \tag{5.14}$$

where k is the number of groups, n_i is the number of observations in the i-th group, $\bar{x}_i$ is the mean of the i-th group, $\bar{x}_{\text{grand}}$ is the grand mean (mean of all observations), N is the total number of observations.

To perform a post-hoc analysis following the ANOVA, the Tukey HSD test was used. This test compares all possible pairs of means to determine if they are significantly different from each other. The formula for Tukey HSD is as follows:

$$HSD = q_{\alpha,k,N-k}\sqrt{\frac{MS_{\text{within}}}{n}} \tag{5.15}$$

$q_{\alpha,k,N-k}$ is the range distribution critical value at the desired significance level (α),for k groups and N−k degrees of freedom.

MS_{within} is the mean square within groups from the ANOVA.

n is the number of observations per group (assuming equal sample sizes).

The test statistic for each pair of group means ($barX_i$ and $barX_j$) is calculated as:

$$\left|\bar{X}_i - \bar{X}_j\right| \tag{5.16}$$

Analysis of Transformed Data: Firstly, the belts with three different pre-set tensions were analyzed, with the tensions being 300 N, 690 N, and 800 N, respectively. From eight different cases, several groups with clear and significant performance were selected, using data from case 1. The data distribution is as Figure 5.3, Figure 5.4.

Use ANOVA and Tukey HSD to quantify the differences among three different distributions. The result of ANOVA is: F=13300.

Table 5.2: Group Comparison Results

group1	**group2**	**mean diff**	**p value**	**reject**
800	390	20.9	7.34e-50	TRUE
800	690	19.8	3.6e-22	TRUE
390	690	-1.10	4.3e-37	TRUE

The ANOVA results show a very significant F-value (13300), indicating that there is a statistically significant difference in the means of at least two groups. This is strong evidence that the means of the distributions of these three groups are not the same after undergoing the Yeo-Johnson transformation. The results of the Tukey HSD test further reveal the specifics of these differences: According to the Tukey HSD test results, all of

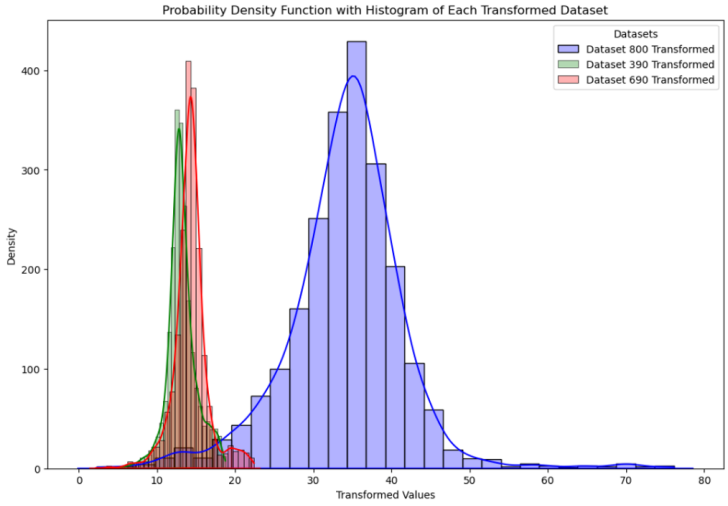

Figure 5.3: PDF in case 1 from transformed data

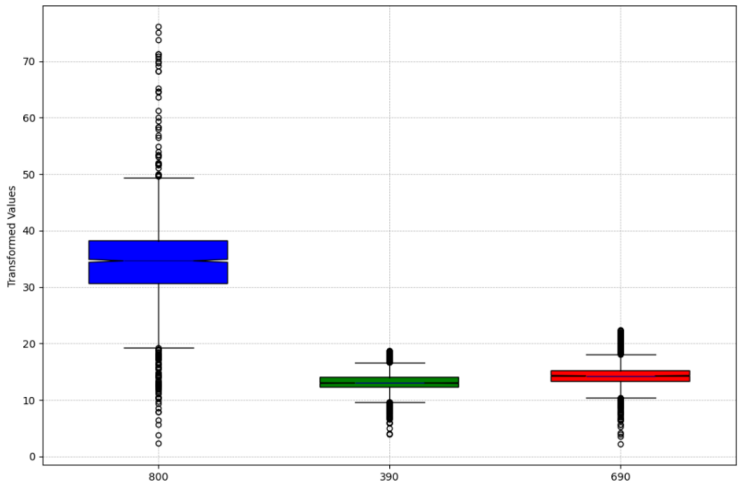

Figure 5.4: Boxplot in case 1 from transformed data

p value are too much smaller than 0.05 which is the experience value in statictical test. That means the different distributions are totally different. The mean difference between the 800 group and the 390 group is 20.9, while the mean difference between the 800 group and the 690 group is 19.8. This means that the difference between the 800 group and the 390 group is slightly larger than the difference between the 800 group and the 690 group, although both are very significant. Therefore, it can be said that the difference between the 800 group and the 390 group is greater.

First, we applied data conversion to process the dataset. While this method helped clarify the discrepancies between the datasets, it is evident that the 800N maximum

tension dataset significantly differs in distribution from the others. However, the trend observed does not support the hypothesis that higher tension leads to smaller time differences. This could be due to alterations in the dataset during the data conversion process, which might not have yielded the desired results. Consequently, it is essential to also explore methods that do not involve data conversion.

Without transforming the data, directly compare the situations under different tension according to the Poisson distribution. From two scenarios, filter out the three different stresses corresponding, selecting only case 1 and case 6. The probability distribution and box plot are as Figure 5.5, 5.6, 5.7, 5.8.

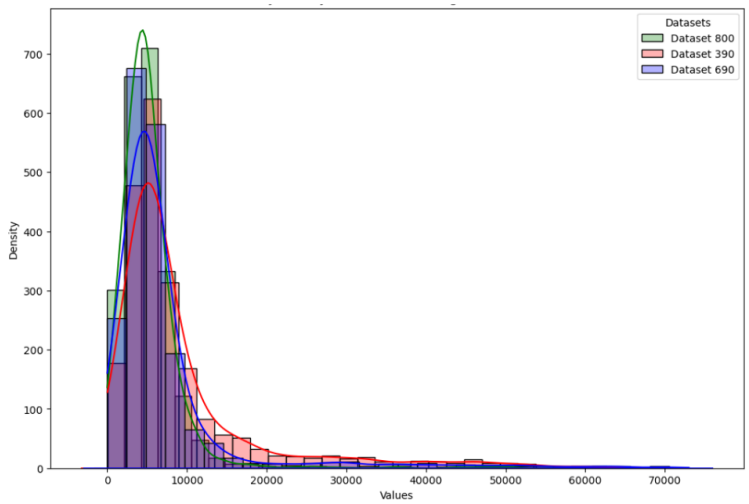

Figure 5.5: PDF in case 1

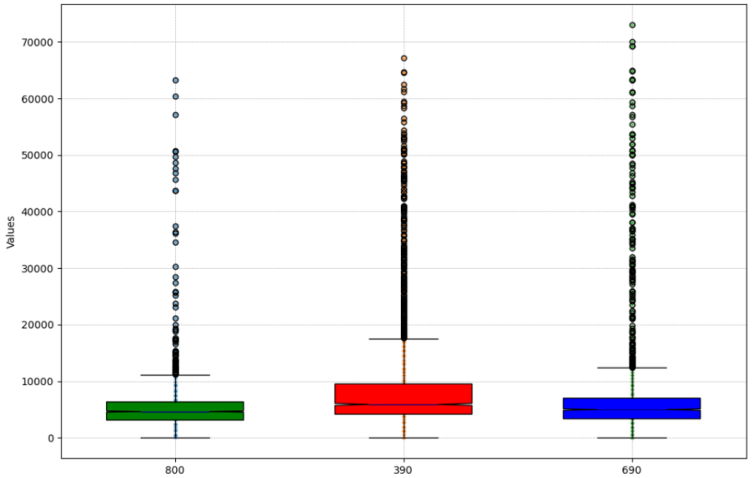

Figure 5.6: Boxplot in case 1

When analyzing non-Gaussian distributions, we utilize the Kruskal-Wallis H test and Dunn's multiple comparison test, followed by Cliff's Delta effect size to quantify the

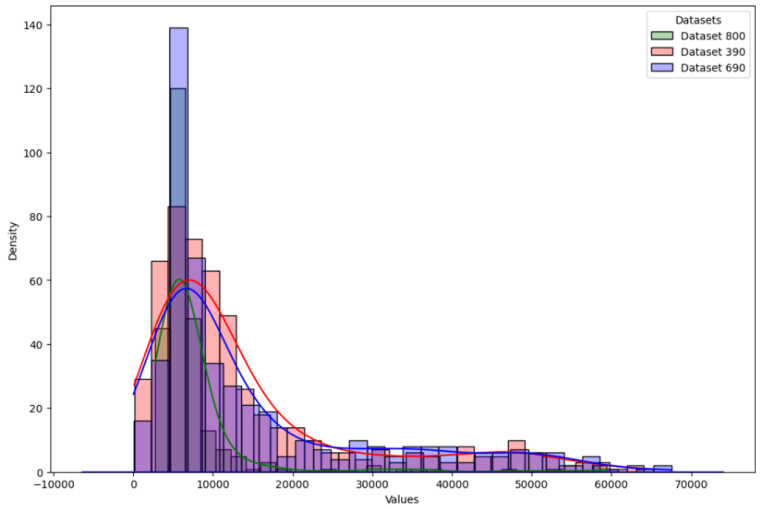

Figure 5.7: PDF in case 6

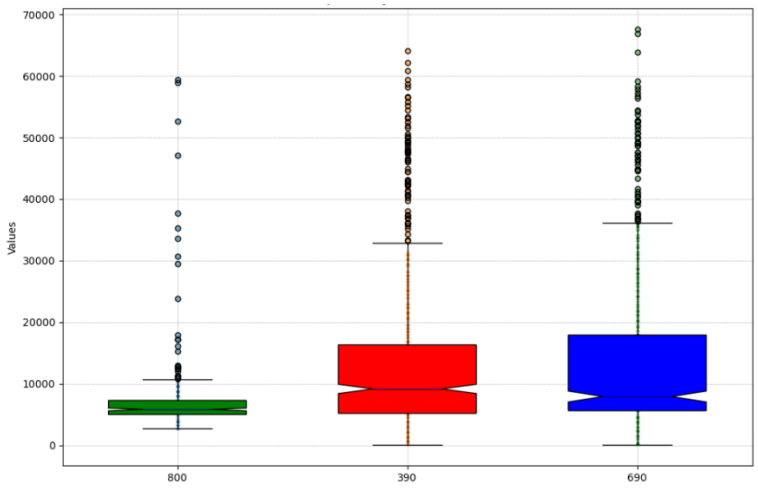

Figure 5.8: Boxplot in case 6

magnitude of differences between distributions.

The Kruskal-Wallis H test is a non-parametric method used to determine whether samples originate from the same distribution, particularly when the assumptions of ANOVA are not satisfied. The formula for the Kruskal-Wallis H test is as follows:

$$H = \left(\frac{12}{N(N+1)} \sum_{i=1}^{k} \frac{R_i^2}{n_i} \right) - 3(N+1) \tag{5.17}$$

N is the total number of observations across all groups. k is the number of groups. R_i is the sum of the ranks for the i-th group. n_i is the number of observations in the i-th group.

Dunn's test is a post-hoc analysis used following a significant Kruskal-Wallis test to determine which specific groups are different. The formula for Dunn's test is:

$$z = \frac{R_i - R_j}{\sqrt{\left(\frac{N(N+1)}{12}\right)\left(\frac{1}{n_i} + \frac{1}{n_j}\right)}} \tag{5.18}$$

R_i and R_j are the mean ranks of the groups being compared. N is the total number of observations. n_i and n_j are the number of observations in the respective groups.The z-value is compared against the critical value from the standard normal distribution to determine significance.

Cliff's delta is a measure of the effect size for ordinal data. It indicates the degree of overlap between two distributions. The formula for Cliff's delta is:

$$\delta = \frac{2(A - B)}{n_1 n_2} \tag{5.19}$$

A is the number of times a value from the first group is greater than a value from the second group. B is the number of times a value from the first group is less than a value from the second group. n_1 and n_2 are the number of observations in the respective groups. Cliff's delta ranges from -1 to 1, 1 indicates all values in the first group are greater than the second group. -1 indicates all values in the first group are less than the second group. 0 indicates no difference between the groups.

The results of the Kruskal-Wallis H test, Dunn's post hoc test for multiple comparisons, and the calculation of Cliff's delta provide an in-depth analysis of the distribution of the three groups of data. Here is a detailed analysis of the results: For Case 1, the result of Kruskal-Wallis H test is :Statistic: 304, P-value: 7.24×10^{-67}. And according to Dunn's multiple comparison test can get the p value between 390 and 690 is 1.4×10^{-26},between 800 and 390 is 4.02×10^{-63},between 800 and 690 is 1.24×10^{-66}.

Table 5.3: Cliff's delta Values for Comparisons Between Groups

Comparison	**Cliff's delta**	**Range**
Group 800 vs. Group 390	-0.299	[-1, 1]
Group 800 vs. Group 690	-0.104	[-1, 1]
Group 390 vs. Group 690	0.193	[-1, 1]

The results of the Kruskal-Wallis H test are highly significant, indicating that at least one group's distribution significantly differs from the others. In other words, this result suggests that at least one of the three groups has a median that significantly differs from the others.

The results of Dunn's post hoc test for multiple comparisons further reveal significant differences between specific groups. Specifically: The P-value between the 390 group and the 690 group is approximately 1.4×10^{-26}, indicating an extremely significant difference. The P-value between the 390 group and the 800 group is about 4.02×10^{-63}, which also shows an extremely significant difference, and this P-value is smaller than that between the 390 and 690 groups, suggesting that the difference between these two groups is even more significant.

The P-value between the 690 group and the 800 group is approximately 1.24×10^{-66}, indicating a significant difference between them as well, but this P-value is relatively larger, meaning the difference is smaller compared to that between the 390 and 800 groups.

The results of Cliff's delta provide another perspective on the size of these differences: Cliff's delta between the 800 group and the 390 group is -0.299, indicating a medium to large effect size, and this difference is the largest among the three comparisons. This means that the difference between the 800 group and the 390 group is not only statistically significant but also substantial in practical terms. Cliff's delta between the 800 group and the 690 group is -0.104, indicating a smaller effect size, showing that the difference between these two groups is smaller than between the 390 and 800 groups. Cliff's delta between the 390 group and the 690 group is 0.193, indicating a medium effect size, but this difference is smaller than between the 800 and 390 groups. In summary, the difference between the 800 group and the 390 group is not only statistically highly significant but also the largest in terms of effect size. This indeed suggests that the difference between the 800 group and the 390 group is greater than between the 800 and 690 groups, as well as between the 390 and 690 groups. This indicates that, in terms of median, the 800 group and the 390 group have distinctly different distributions in their characteristics. For case 6 the result of Kruskal-Wallis H test is 72.3, P-value is 1.93×10^{-16}. The result of Dunn's Multiple Comparison test is : p value between 390 and 690 is 1, between 690 and 800 is 5.42×10^{-15}, between 390 and 800 is 1.36×10^{-13}. And Cliff's Delta value is as Table 5.4.

Table 5.4: Cliff's Delta Values for Comparisons Between Groups

Comparison	**Cliff's Delta**	**Range**
Group 800 vs. Group 390	-0.324	[-1, 1]
Group 800 vs. Group 690	-0.364	[-1, 1]
Group 390 vs. Group 690	-0.021	[-1, 1]

The Kruskal-Wallis H test shows significant differences among the three groups. Dunn's test indicates no significant difference between 390 and 690, but clear differences between the other two groups. According to Cliff's delta, the difference between 800 and 690 is greater. The result is similar to the one in Case 1. Not only does this method effectively distinguish between the three groups, but it also uses effect size to illustrate the magnitude of differences in data distribution. The trend of the results aligns with the initial hypothesis of this study, laying the groundwork for further analysis in the next phase. Finally, the next phase will also employ a non-data transformation approach to handle the data.

5.3 System On-Condition Observation and Fault Detection in Laboratory Environments

A fault detection method is proposed based on the above data analysis, which reveals that the simulated data exhibits some differences. From the earlier TRL stages of product design, it is understood that the larger the belt tension, the smaller the time difference data obtained by the encoder. The estimated fault tension value should be around 500 N.

The on-site implementation steps for system status detection and on-condition monitoring involve measuring the belt tension at the initial operation stage of the equipment and then collecting the time difference data detected by the encoder over a period of time. The equipment will operate for several days or months, periodically capturing a segment of time difference data as a dataset for analysis to estimate the current state. If the value exceeds a certain range, it is considered a fault.

To conduct the fault detection experiment, three different tension data points from Case 1 were selected. First, it is ensured that the differences in data distribution can be described using statistical tests, meaning that the data corresponding to each tension is stable within a certain range under the assumed tension. D.Valis used a local linear model to process the belt experience data, resulting in a technically reliable estimated trend [4]. The local linear model model includes two processing steps: first, using the Kalman filter to estimate the data, and then using the Kalman smoother to smooth the estimated curve, clearly showing the trend of the overall continuous data. In this study, local linear model processing was performed for all three tensions. Since the time difference data contains one variable, a one-dimensional model is suitable.

First, the initial covariance P0 is chosen, which reflects the uncertainty of our initial state estimate. If the initial state estimate is very uncertain, a larger P0 is selected. In this way, the filter will rely more on the measurements in the initial state. If the initial state estimate is relatively certain, a smaller P0 is selected. Thus, the filter will rely more on the initial estimate in the initial state. Given the data appears to fluctuate widely and spans a broad range, a moderate value, such as P0=1, can be chosen to balance the uncertainty of the initial estimate.

The process noise covariance Q reflects the uncertainty in the system model. If the uncertainty in the system model is large, a larger Q is chosen. This way, the filter will rely more on the measurements rather than the model predictions. If the uncertainty in the system model is small, a smaller Q is chosen. This way, the filter will rely more on the model predictions. Based on the data, if the variations in the data are relatively smooth, a smaller value can be chosen, such as $Q = 1 \times 10^{-5}$.

The measurement noise covariance R reflects the uncertainty in the measurements. If the measurement noise is large, a larger R is chosen. This way, the filter will rely more on the model predictions rather than the measurements. If the measurement noise is small, a smaller R is chosen. This way, the filter will rely more on the measurements. Based on the data, if the measurement noise is relatively small, a moderate value can be chosen, such as $R = 1 \times 10^{-2}$.

Then, these parameters are applied to the Kalman filter and the Kalman smoother.

The prediction formula for the Kalman filter is as follows, State prediction:

$$\hat{x}_{k|k-1} = \hat{x}_{k-1|k-1} \tag{5.20}$$

Covariance prediction:

$$P_{k|k-1} = \mathbf{P}_{k-1|k-1} + \mathbf{Q} \tag{5.21}$$

Kalman Gain:

$$K_k = \frac{P_{k|k-1}}{P_{k|k-1} + R} \tag{5.22}$$

State update:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(z_k - \hat{x}_{k|k-1}) \tag{5.23}$$

Covariance update:

$$P_{k|k} = (1 - K_k)P_{k|k-1} \tag{5.24}$$

Smoother Gain:

$$J_k = \frac{P_{k|k}}{P_{k+1|k}} \tag{5.25}$$

Then, the state and covariance are smoothed using the following formulas:

$$\hat{x}_{k|N} = \hat{x}_{k|k} + J_k \left(\hat{x}_{k+1|N} - \hat{x}_{k+1|k} \right) \tag{5.26}$$

$$P_{k|N} = P_{k|k} + J_k \left(P_{k+1|N} - P_{k+1|k} \right) J_k \tag{5.27}$$

The final graphs for the three different tensions are as Figure 5.7 and Figure 5.8 and Figure 5.9.

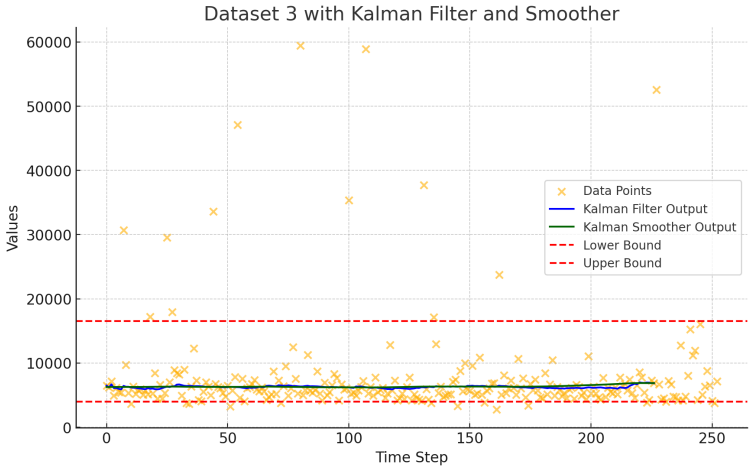

Figure 5.9: Data of 800 N with Kalman filter and smoother

It can be seen that the distribution of each dataset corresponding to different tensions is stable, with no obvious internal trends. This allows for the combined analysis of the datasets and the use of statistical tests to find differences, thereby identifying the system's state.

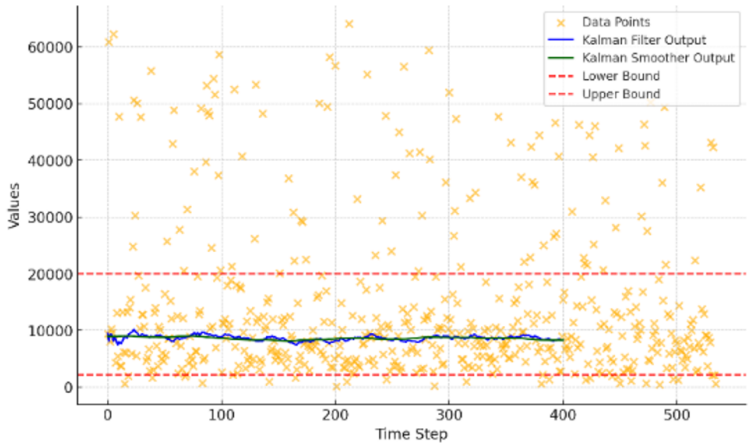

Figure 5.10: Data of 690 N with Kalman filter and smoother

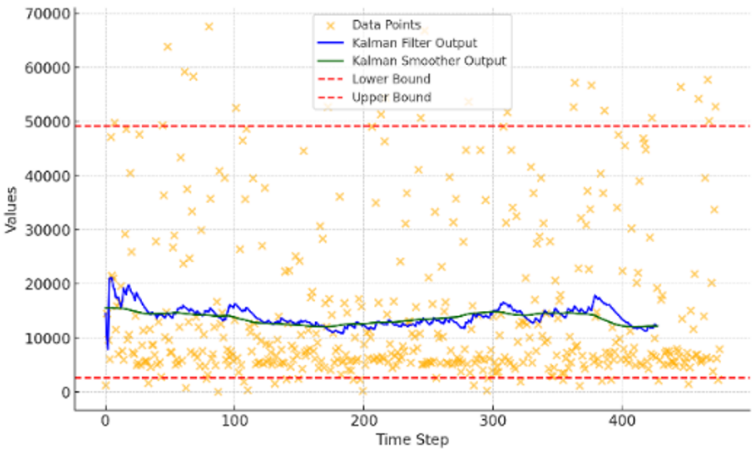

Figure 5.11: Data of 390 N with Kalman filter and smoother

The statistical effect size from the analysis of the data in Case 1 can describe the differences in the time difference data corresponding to the three tensions. It was found that the time difference data for 390 N is statistically larger than those for 690 N and 800 N. The absolute value of the effect size between 390 N and 800 N is greater than that between 800 N and 690 N, and the statistical effect size between 690 N and 800 N is smaller than that between 390 N and 690 N. In other words, the time difference distribution of the three tensions shows a trend opposite to the tensions themselves under statistical testing.

Therefore, it can be concluded that if the tension of 800 N is used as the initial comparison tension and the Cliff's Delta effect size is between -0.32 and -0.36, then the belt's state is approaching the fault interval. However, using this as a standard cannot precisely indicate a fault, and additional fault detection steps are necessary.

Artur Pollak used the autoencoder for fault detection in maintaining belt drive systems [1].The autoencoder is a type of neural network known for its excellent memory and comparison capabilities as Figure 5.10, making it useful for distinguishing multiple deviation signals or data. The autoencoder is a generative unsupervised deep learning algorithm that applies back propagation to perform dimensionality reduction by using the target data as input values. It consists of three components: the encoder, the coder, and the decoder. The autoencoder works by compressing the input data with the encoder, generating a code with the coder, and then reconstructing the input data as accurately as possible with the decoder. If there is a data point x belonging to R^d, the reconstruction error between the training data and the autoencoder output will be used as the loss function:

$$L(x, \hat{x}) = \|x - \hat{x}\|^2 \tag{5.28}$$

x is the original input data, $\hat{x}$ is the reconstructed data. $\|\cdot\|$ denotes the norm, typically the Euclidean norm This loss function helps the autoencoder learn to reconstruct the input data by minimizing the reconstruction error, making it effective for anomaly detection when applied to data from belt drive systems.

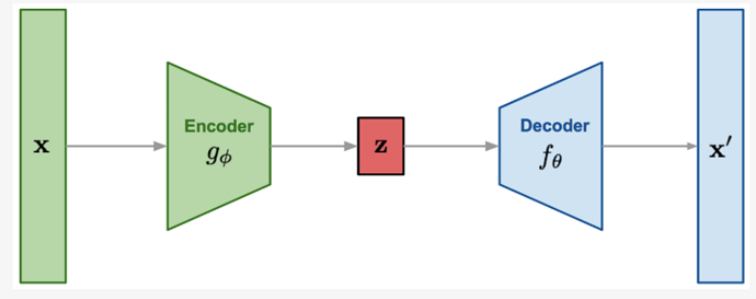

Figure 5.12: The autoecoder Architecture

After training, the autoencoder will reconstruct other normal datasets and fault datasets. Those with higher reconstruction errors will be considered anomalies, as only normal data is used to train the autoencoder.

In Artur Pollak's experimental data, fault data and normal data were well distinguished. However, based on previous observations, the differences between the datasets in this study are minimal, making the performance of a simple autoencoder unreliable. Considering that a composite model will perform better and distinguish the two types of data more effectively, a more complex approach is resonable.

Peihua Han and others conducted fault detection research in the field of ship maintenance using a combined model of variational autoencoder(VAE) and long short-term memory(LSTM) [6]. VAE is a variant of the autoencoder based on Bayesian theory [10]. The purpose of VAE is not only to reconstruct data but also to regularize the latent space distribution to be close to a prior distribution. VAE replaces the encoded variable Z of the autoencoder, as shown in the Figure 5.11.

The encoder first maps the input data to a probability distribution in the latent space, typically using a normal distribution with a mean of 0 and a standard deviation of 1 as the latent probability distribution. The encoder generates not z but the mean and

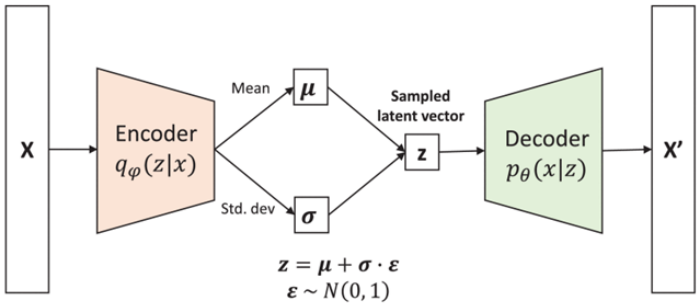

Figure 5.13: VAE Architecture

variance of the distribution $q_\phi(z \mid x)$, as the mean and standard deviation are sufficient to describe a distribution. The decoder samples from the latent distribution to obtain the data corresponding to the latent variable z and maps it back to the original data space to generate the distribution $p_\theta(x \mid z)$, ultimately producing the reconstructed data x'. The training objective of the VAE is to maximize the evidence lower bound(ELBO), which consists of the reconstruction error and the kullback-leibler(KL) divergence. The reconstruction error is the difference between the reconstructed data x' and the original data x, typically measured using the log-likelihood estimate.The KL divergence measures the difference between the latent distribution $q_\phi(z|x)$ produced by the encoder and the prior distribution p(z):$-\mathrm{KL}(q(z|x)\|p(z))$, then Combining these two components, the loss function for the VAE is:

$$\mathcal{L}_{\text{vae}} = -D_{KL}(q_\phi(z|x)\|p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] \leq \log p(x) \tag{5.29}$$

Subsequently, back propagation is used to update the parameters of the encoder and decoder based on the gradients of the loss function. The gradient descent optimization algorithm is employed to adjust the parameters, minimizing the loss function. This involves calculating the gradients of the loss function with respect to the encoder and decoder parameters and iteratively updating these parameters in the direction that reduces the loss. The overall process ensures that the VAE learns to effectively encode the input data into a latent representation and accurately reconstruct the data from this representation, while maintaining the regularization imposed by the KL divergence.

LSTM is a type of RNN, but it incorporates a memory cell to regulate the flow of information within the unit, enhancing the standard RNN architecture [6]. As shown in the Figure 5.12.

Each memory cell contains three nonlinear gating units, which function to protect and regulate the cell's state. LSTM can use these gating units to address the vanishing gradient problem. For each element in the input sequence, LSTM uses the following formulas:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \tag{5.30}$$

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \tag{5.31}$$

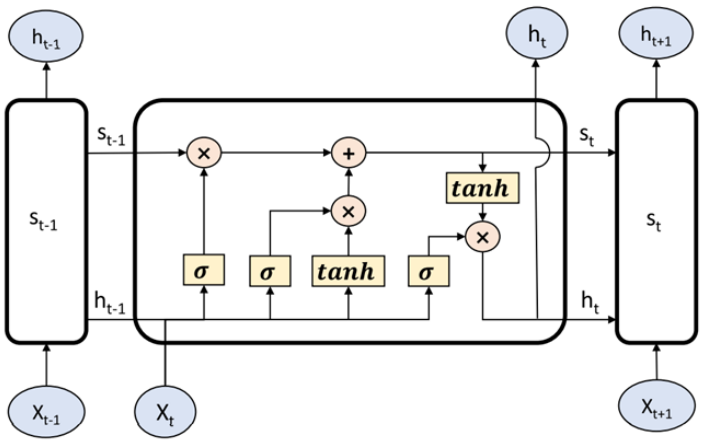

Figure 5.14: LSTM Cell Architecture

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \tag{5.32}$$

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \tag{5.33}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{5.34}$$

$$h_t = o_t \odot \tanh(c_t) \tag{5.35}$$

At time t, h_t represents the hidden state, c_t represents the cell state, and x_t represents the input. h_{t-1} and c_{t-1} represent the hidden state and cell state at time t-1 respectively. i_t, f_t, g_t, and o_t are the input gates, forget gate, cell gate, and output gate respectively. The cell gate and output gate use the sigmoid function, where σ is the sigmoid function and $\odot$ represents the Hadamard product (element-wise multiplication). W and b are the weights and biases in the LSTM unit.

This thesis introduces a Variational Autoencoder based on long short term memory and variational autoencoder (LSTM-VAE). The LSTM-VAE is a combination of VAE and LSTM, where VAE assumes that the data stream is independent and identically distributed over time. To introduce time dependency into this model, replace the feed-forward network in the VAE with LSTM. Figure 5.13 shows the structure of LSTM-VAE unfolded in the time dimension.

Given the multivariate input x_t at time t, the encoder LSTM outputs the hidden state h_t using x_t, h_{t-1}, and c_{t-1}. Then, h_t is fed into two linear modules to estimate the mean μ_t and the log variance of the posterior $p(z_t|x_t)$. After randomly sampling z_t from $p(z_t|x_t)$, it is input into the decoder LSTM. Finally, a linear module outputs the reconstructed input $\hat{x}$. The parameters ϕ and θ of the encoder and decoder are obtained by minimizing the following loss function:

$$\text{Loss} = \sum_{t=1}^{T} [D_{KL}(q_\phi(z_t|x_t) \| p_\theta(z_t)) + \text{MSE}(x_t, \hat{x}_t)] \tag{5.36}$$

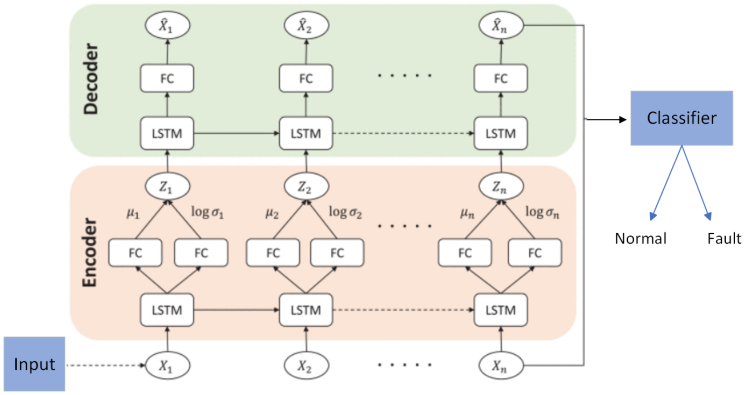

Figure 5.15: Variational Autoencoder with LSTM for Fault Detection

Here, mean squared error(MSE) and T is the length of the sequence. A standard normal distribution N(0,1) is used as the prior for the latent space $p_\theta(z_t)$.

For this experiment with Linear Belt-Driven Motion Systems, the 800 N tension data was used as the training dataset. Using cross-validation, the 800 N dataset was divided into 70 percent for training and 30 percent for validation. Then, the complete datasets of three different tensions were used to test the model. The reconstructed outputs were obtained for each dataset, and the root mean square error(RMSE) for each data point was calculated by comparing the reconstructed output with the original dataset. Finally, all the RMSE values were plotted on the same coordinate system to compare the performance of the three different tensions, as shown in the figure 5.14.

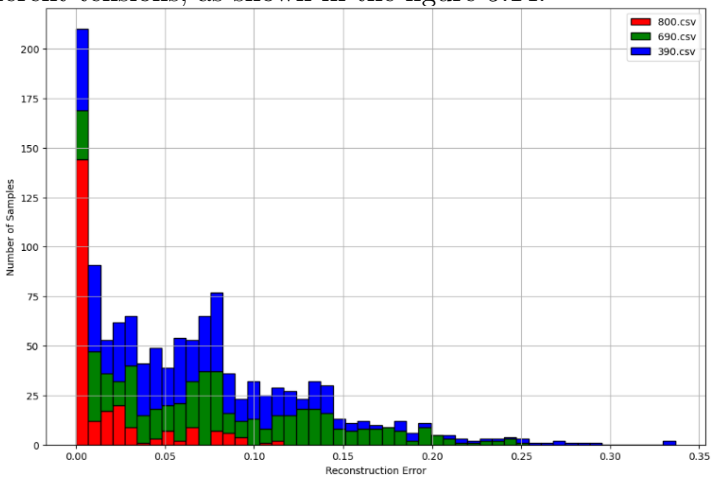

Figure 5.16: Reconstruction Error Distribution for Different Dataset

It can be observed that the RMSE distribution of 390 N differs from the other two. Based on this RMSE data, a classifier is created. The RMSE data for the fault condition

(i.e., 390 N tension) is labeled as class 1, while the RMSE data for the normal conditions (i.e., 800 N and 690 N tensions) is labeled as class 0.

Considering the existence of different classifiers, we will compare multiple classifiers in this experiment. For all classifiers, X represents the standardized reconstruction errors, and y represents the labels corresponding to class 0 and class 1. The specific classifiers are as follows:

The goal of logistic regression(LR) is to find a probabilistic model to predict the probability of the output y given the input X. The model's prediction function is:

$$\hat{y} = \sigma(wX + b) \tag{5.37}$$

In this context, σ is the sigmoid function, which converts any real-valued number into a probability value in the range (0, 1). W is the model parameter (weights), b is the bias.

k-nearest neighbors(KNN) classifies the test data by calculating the distance between the test data and each training data point, selecting the KNN nearest training samples, and voting to determine the label of the test data. The prediction formula is as follows:

$$\hat{y} = \text{mode}\{y_i : x_i \in \text{nearest } K\} \tag{5.38}$$

The nearest K refers to the K training samples that are closest to x, where the distance is measured using the Euclidean distance. *haty* is the predicted label.

A decision tree constructs a tree-structured classifier by recursively selecting the optimal features and splitting the data. The splitting criterion of a decision tree is based on information gain, aiming to find the best split point that maximizes the ability to distinguish between different classes. For binary classification problems, the tree tests at each node when $X[j] \leq$ threshold goes left branch otherwise goes right. Here, j is the feature index, and the threshold is the optimal split point found through the training data.

SVM classifies data by finding a hyperplane that maximizes the margin between the training data points of different classes. For linear SVM, the classification decision is based on:

$$\hat{y} = \text{sign}(wX + b) \tag{5.39}$$

Here, w is the normal vector to the hyperplane, and b is the bias term.

naive bayes(NB) classifier is based on Bayes's theorem and assumes that the input features are mutually independent. The prediction model is as follows:

$$\hat{y} = \arg\max_c P(y = c) \prod_i P(X_i | y = c) \tag{5.40}$$

The comparison of the metrics for all classification models is presented in the table 5.6.

First, the reconstruction error data for normal tension is classified as class 0, and the fault data is classified as class 1. Then, the reconstruction error data for all three tensions is combined and randomly shuffled. Using cross-validation, the data is split into 70 percent for training and 30 percent for validation. The validation set is then used to calculate the performance metrics, and the results are presented in the table. In the table, "support" refers to the number of occurrences for each condition. The metrics are calculated twice:

Table 5.5: Summary of Model Performance Metrics

Metric	**LR**	**KNN**	**DT**	**SVM**	**NB**
Support label 1	218	218	218	218	218
Support label 0	154	154	154	154	154
Support total	372	372	372	372	372
Accuracy	0.58	0.59	0.62	0.59	0.59
Precision 0	0.59	0.66	0.69	0.64	0.59
Precision 1	0.47	0.51	0.53	0.51	0.76
Recall 0	0.96	0.63	0.50	0.68	0.43
Recall 1	0.05	0.53	0.59	0.43	0.74
F1-score 0	0.73	0.64	0.59	0.66	0.51
F1-score 1	0.09	0.52	0.56	0.46	0.74
Macro avg precision	0.53	0.58	0.61	0.57	0.29
Macro avg recall	0.51	0.58	0.55	0.57	0.57
Macro avg f1-score	0.41	0.59	0.61	0.56	0.37
Weighted avg precision	0.54	0.58	0.61	0.59	0.58
Weighted avg recall	0.58	0.59	0.62	0.59	0.59
Weighted avg f1-score	0.47	0.59	0.62	0.58	0.43

once with class 1 as the positive class and once with class 0 as the positive class. This allows for different precision, recall, and F1-score values to be obtained. Based on these, the macro average and weighted average can be calculated.

The positive and negative classes are further categorized into: true positive(TP), true negative(TN), false positive(FP), flase negative(FN).

Accuracy represents the proportion of correct predictions made by the model, including both positive and negative classes:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.41}$$

The DT has the highest accuracy at 0.62, indicating that it has the best prediction accuracy among all samples. The others perform slightly lower than DT.

Precision represents the proportion of samples predicted as positive by the model that are actually positive. Depending on which class is considered the positive class, there are two types of precision:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5.42}$$

Similar to Accuracy, DT has the highest precision for both classes, indicating that it makes fewer errors when predicting the positive class. NB has a precision of 0 for class 1, indicating that this model fails and cannot correctly identify fault conditions.

Recall represents the proportion of actual positive samples that are correctly predicted as positive by the model. There are two types of recall, depending on which class is

considered the positive class:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5.43}$$

Although NB has a recall of 1, it also indicates that this model fails. Additionally, DT has the highest recall for class 1, indicating that it is the most capable in identifying fault samples. LR has the highest recall for class 0 but very low recall for class 1, indicating that it has a strong ability to identify normal samples but cannot identify fault samples.

F1-score is the harmonic mean of precision and recall:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.44}$$

NB has the highest F1-score for class 0, but it is 0 for class 1, indicating that it performs well in recognizing normal samples but is completely ineffective for fault samples. DT has the highest F1-score for class 1, indicating that it has the best overall capability in recognizing fault samples.

Macro average is the arithmetic mean of each class's metrics, without considering the differences in the number of samples for each class:

$$\text{Macro Precision} = \frac{\sum \text{Precision}}{N} \tag{5.45}$$

$$\text{Macro Recall} = \frac{\sum \text{Recall}}{N} \tag{5.46}$$

$$\text{Macro F1-Score} = \frac{\sum \text{F1-Score}}{N} \tag{5.47}$$

where N is the total number of classes.

Weighted average takes into account the number of samples in each class, performing a weighted average of each class's metrics based on the number of samples:

$$\text{Weighted Precision} = \frac{\sum(\text{Precision} \times \text{Support})}{\sum \text{Support}} \tag{5.48}$$

$$\text{Weighted Recall} = \frac{\sum(\text{Recall} \times \text{Support})}{\sum \text{Support}} \tag{5.49}$$

$$\text{Weighted F1-Score} = \frac{\sum(\text{F1-Score} \times \text{Support})}{\sum \text{Support}} \tag{5.50}$$

Where "support" corresponds to the number of samples in each class, and the quantity is used as the weight to perform weighted multiplication with the corresponding metrics.

Macro average and weighted average show similar performance. DT performs the best in both metrics, indicating that its overall performance is the best regardless of whether class balance is considered.

In summary, DT is the relatively best classifier in terms of overall performance. It outperforms other classifiers in all metrics, particularly in accuracy, macro average, and weighted average. The TRL 4-5 stage involves testing a small-scale prototype. The results indicate that the classification is not very effective because the data collection process

is uncontrollable, and the small size may result in a less clear relationship between the time difference data and belt tension trends. In the later stages, data will be collected from larger prototypes, and the data will undergo the same processing and analysis. In the next phase, the data collection process will be optimized and adjusted.

Chapter 6

Experiments for Technology Readiness Level 5 to 6 Demonstrations and Result

6.1 Data Processing in a Related Environment

In the TRL 5-6 stage, the linear belt-driven motion system will use a larger prototype and be tested in relevant environments. Additionally, the loading and operating conditions of the belt system will be more standardized before data collection. In this experiment, the belt system will have 24 loaded carriages, each carrying a one-kilogram weight, and the initial speed of the belt will be set to $4\,\mathrm{m/s^2}$. The preset tensions used are 800 N, 600 N, and 470 N.

Under these preset conditions, data will still be collected using an encoder. There will be 8 cases, and after collecting the data, the probability density distribution plots for each of the 8 cases will be created. The case with the most significant difference will be selected for analysis. Case 3 was ultimately chosen for analysis, and the probability density distribution plot is shown as Figure 6.1.

It can be seen that there are certain differences in the performance across the three datasets. The results are based on the original data, without any transformation. This outcome provides a foundation for subsequent data analysis and model development.

6.2 Data Analysis in a Related Environment

First, statistical tests are used to determine if there are significant differences among the three datasets. The result from the Kruskal-Wallis H Test is: H-statistic = 7800. This result indicates that the median of at least one group is significantly different from the others. The Kruskal-Wallis H Test is a non-parametric test used to determine if there are significant differences in the overall distribution among three or more independent samples.

Cliff's Delta is an effect size measure used to describe the magnitude of the difference between the distributions of two groups. The values of Cliff's Delta range from -1 to $+1$.

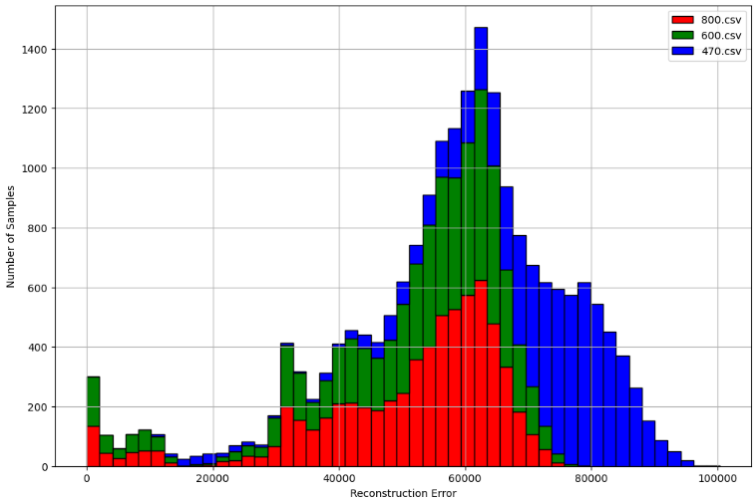

Figure 6.1: PDF Comparison within TRL 6

The larger the absolute value, the greater the difference. The Cliff's Delta values between 470 and 600, and between 470 and 800, are close to 0.76, indicating a large effect size and a significant difference between these datasets. In contrast, the Cliff's Delta value between 600 and 800 is only 0.01, indicating almost no difference between these two datasets.

6.3 Fault Detection in Related Environments

Since the belt tension corresponding to each dataset is a preset parameter before the belt system operates, the tension will continuously decrease due to belt wear during the operation of the belt system. However, because the state of the belt will not change significantly in a short time, the change in time difference data will not be substantial. Therefore, it is necessary to verify whether the time difference data in the datasets corresponding to the different tensions do not have significant trend changes.

Scatter plots of the data for the three different tensions are drawn, and the Kalman filter and Kalman smoother are used to estimate their trends. This allows the use of all the data corresponding to each tension in subsequent analysis and modeling.

The results for the three datasets are shown in Figure 6.2, Figure 6.3 and Figure 6.4.

Observing the distribution and fit of all three datasets confirms that the data points do not show significant trend changes under the current preset tension conditions. Therefore, the overall data can be analyzed in subsequent stages.

As in the previous stage, the LSTM-VAE model is used for modeling. The distribution plot of the reconstruction error values obtained is shown as Figure 6.5.

The red dashed lines in the reconstruction error distribution plots represent the different tension conditions. Observing these plots, it can be seen in table 6.3.

The reconstruction error for the 470 N tension dataset shows significant deviations

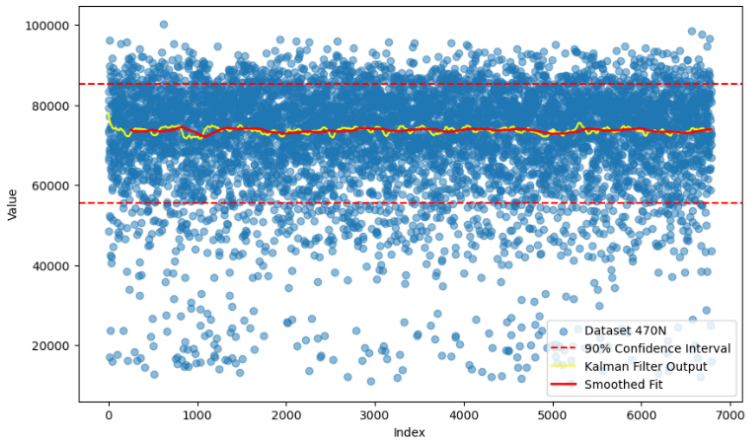

Figure 6.2: Dataset 470 N with Kalman Smoother Fit and Confidence Interval within TRL 6

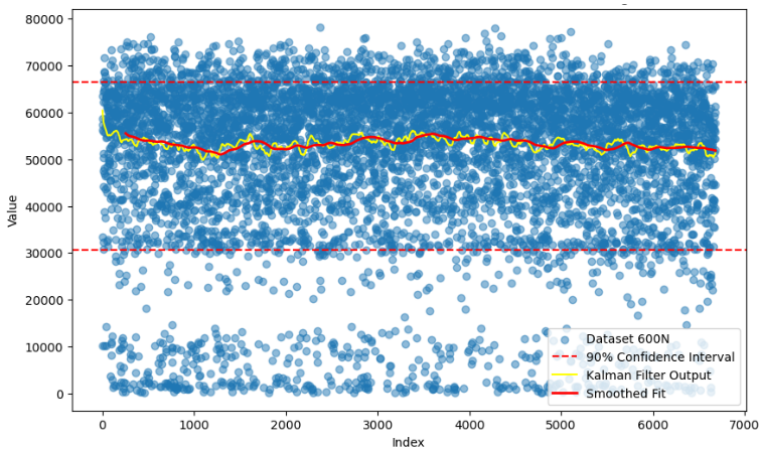

Figure 6.3: Dataset 600 N with Kalman Smoother Fit and Confidence Interval within TRL 6

compared to the 600 N and 800 N tension datasets.

The 600 N and 800 N tension datasets have similar distributions, indicating similar behavior under these conditions.

The LSTM-VAE model effectively highlights the differences in reconstruction errors for varying tensions, allowing for better analysis and fault detection in the belt-driven motion system. Subsequently, classifiers are built using the current reconstruction error data, including LR, KNN, DT, SVM, and NB. The results are shown in the table below:

From the table, it can be seen that: SVM shows the highest accuracy (0.884), indicating it classifies most accurately overall. For label 0, SVM exhibits the highest precision (0.94) among all models. For label 1, LR, KNN, and SVM show similar and relatively high performance. In terms of recall, SVM performs best for label 1 but less well for label

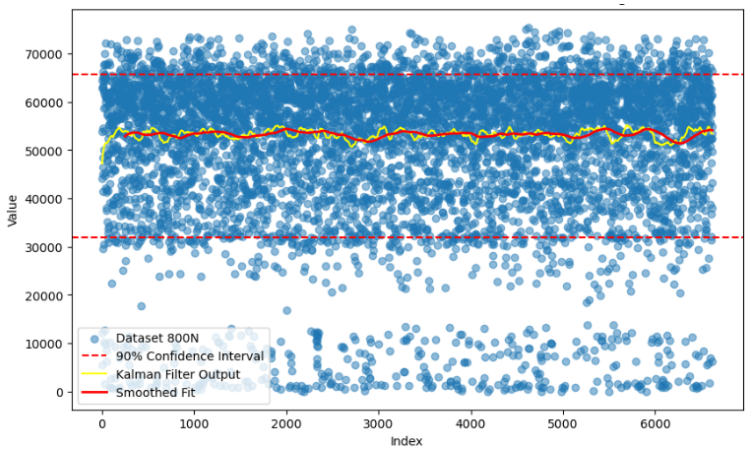

Figure 6.4: Dataset 800 N with Kalman Smoother Fit and Confidence Interval within TRL 6

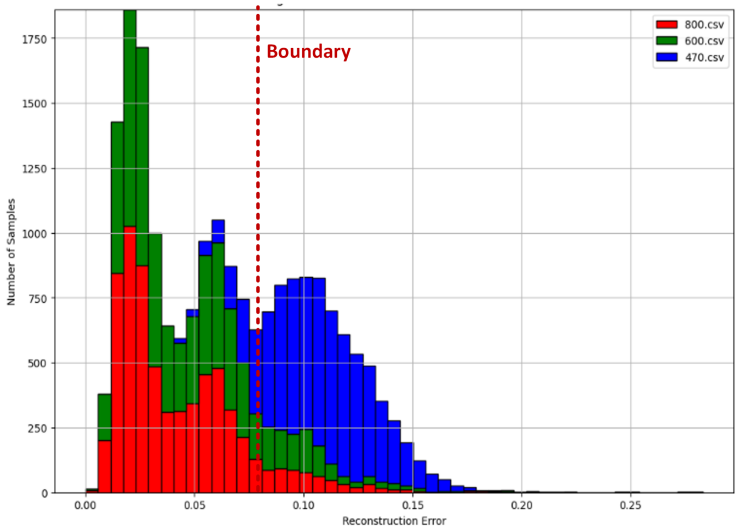

Figure 6.5: Reconstruction Error Distribution for Datasets 800 N, 600 N, and 470 N

0. SVM has very high F1-scores for both classes, especially for label 1 (0.84), indicating a good balance between precision and recall. SVM also excels in Macro Average and Weighted Average, reflecting its efficient handling of different classes.

NB and LR usually perform well with linearly separable data, as evidenced by their accuracy, macro average, and weighted average metrics. KNN perform well in terms of precision and recall, has its performance influenced by the choice of k-value and distance computation method.

Ultimately, SVM will be chosen as the classifier due to its overall superior performance. It shows high performance across most metrics, likely because it effectively classifies by

Table 6.1: Summary of Model Performance Metrics

Metric	**LR**	**KNN**	**DT**	**SVM**	**NB**
Support 0	3978	3978	3978	3978	3978
Support 1	2042	2042	2042	2042	2042
Support total	6020	6020	6020	6020	6020
Accuracy	0.87	0.86	0.83	0.88	0.87
Precision 0	0.90	0.91	0.87	0.94	0.91
Precision 1	0.82	0.79	0.75	0.79	0.81
Recall 0	0.91	0.88	0.87	0.88	0.90
Recall 1	0.81	0.84	0.75	0.90	0.82
F1-score 0	0.90	0.90	0.87	0.91	0.90
F1-score 1	0.81	0.81	0.75	0.84	0.82
Macro avg precision	0.86	0.85	0.81	0.87	0.86
Macro avg recall	0.86	0.86	0.81	0.89	0.86
Macro avg f1-score	0.86	0.86	0.81	0.88	0.86
Weighted avg precision	0.87	0.87	0.83	0.89	0.87
Weighted avg recall	0.87	0.87	0.83	0.88	0.87
Weighted avg f1-score	0.87	0.87	0.83	0.89	0.87

maximizing the margin between decision boundaries.

Chapter 7

Integration Into Smart Hinge System and Graphical User Interface

7.1 Comprehensive Analysis of Smart Edge Products and Introduction to Integrated Data Processing Algorithms

At the initial stage of the entire data processing workflow, encoders collect data through a real-time sampling system. This data is first transmitted to servers operating on the Linux system as shown in Figure 7.1. Each time the device is activated, these raw data are automatically appended to a .log file, which is updated in real-time within the Linux environment, preserving the data's originality and integrity. This method of continuous data recording lays a solid foundation for subsequent data analysis.

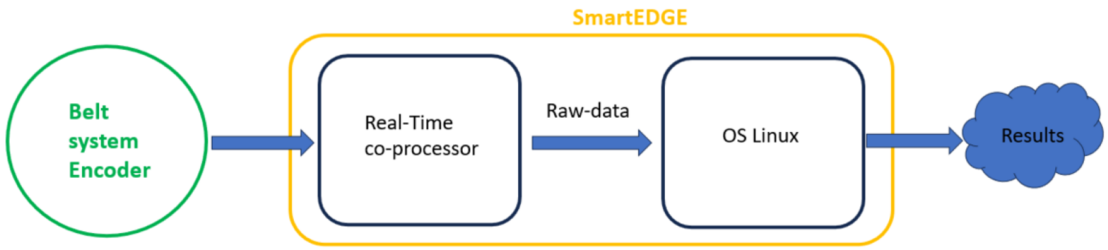

Figure 7.1: Encoder Overall Workflow Diagram

While data is continuously collected, the .log file is constantly updated. The Routine.py program runs within the OS Linux environment, monitoring changes in the .log file in real time as illustrated in Figure 7.2. The primary task of this program is to extract new data entries from the updated .log file and distribute them to the appropriate analysis software according to requirements. The essence of this process lies in the real-time nature and accuracy of the data, ensuring that each update is processed promptly.

Upon detecting updates in the .log file, Routine.py immediately notifies the analysis

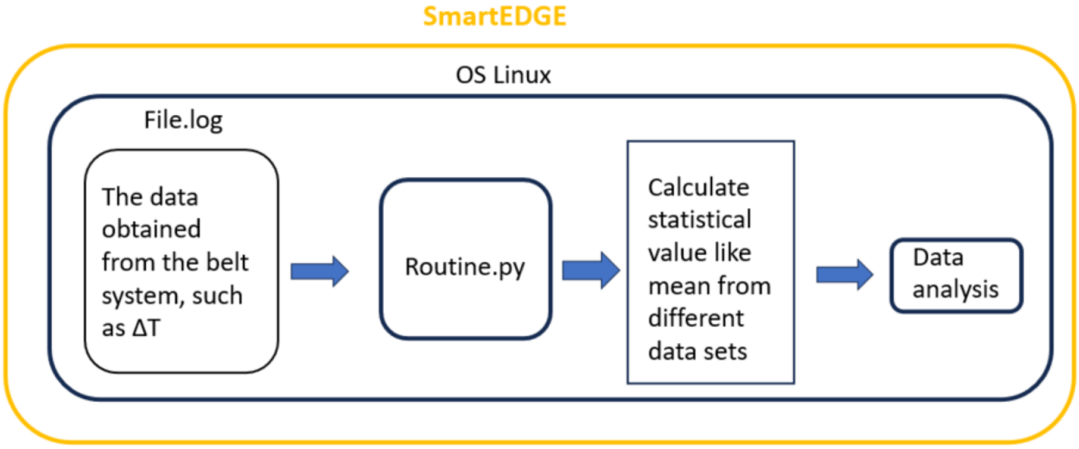

Figure 7.2: SmartEDGE Data Analysis Workflow

program via a predefined queue mechanism. This queue-based communication mechanism ensures the orderly and timely processing of data. Once the data is received by the analysis program, it promptly undergoes processing, with each program executing algorithms tailored to specific data characteristics, outputting the results of the analysis. Given that the ultimate product of this thesis focuses solely on statistical analysis of data distribution for edge computing processes, the TRL 6 stage concentrates specifically on the time difference between active and passive wheels as shown in Figure 7.3. The software analyzes the input data at each step of operation and generates plot points that conform to a Gaussian distribution. After accumulating a certain sample size (for example, 24 hours of operational data), these plot points can be used to extract critical information about the belt tension state and ultimately generate a comprehensive "point" that detailedly depicts the current operational status of the equipment.

7.2 Development of Graphical User Interface for Operation and Maintenance Status

In conclusion, this thesis also makes contributions toward improving a GUI interface, incorporating previously analyzed data describing various statistical distribution parameters into the cloud server, and connecting it to machinery within the factory. This setup allows the server to receive information from the machines and display it on the GUI interface in real-time. As shown in Figure 7.4, the interface allows users to set a time range and obtain corresponding data curves, where each data point represents an average of 24 data points corresponding to 24 different carts. If the number of carts is changed in future experiments, the average capacity corresponding to these points will also vary. Different colors on the interface represent different cases. Due to the instability of the results in the last four cases, only the first four are chosen. This approach allows for the

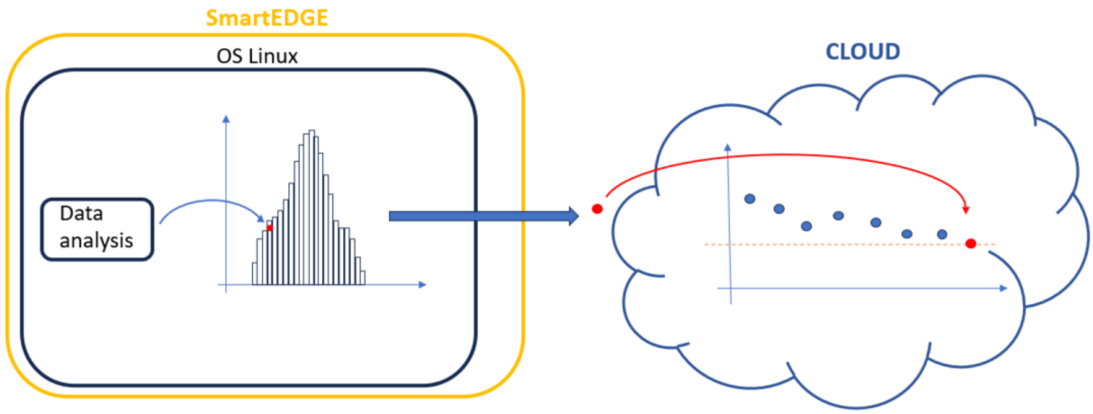

Figure 7.3: SmartEDGE to Cloud Data Transfer and Analysis

real-time observation of the system's status by plotting the average data curve. In the plot below, Case 4 is not displayed because the encoder collected too few data points for this case during the current time period.

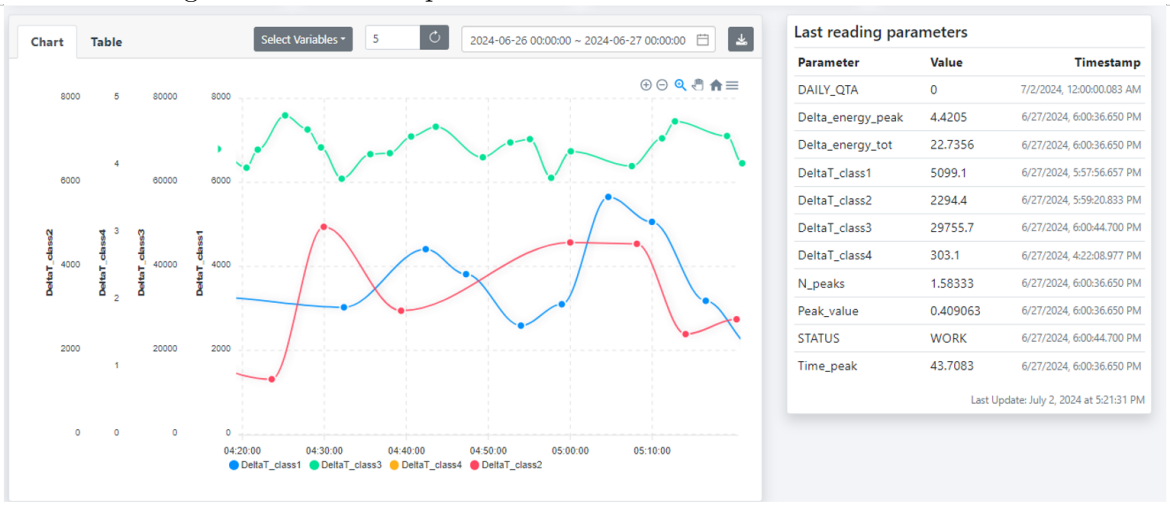

Figure 7.4: Visualization of Belt System Data with Parameter Readings

Additionally, another display interface focuses on the average data corresponding to each time point, as depicted in Figure 7.5. Since the encoder's data collection across the eight different cases occurs randomly, the interval between each data point within a case is not uniform. The data collection on the right, labeled "Most Recent Data Collection," refers to all the information corresponding to the last observed data point. It should be noted that this thesis only added the Table page, along with functionalities and data related to the time difference deltaT; other data and interfaces are not pertinent to this study.

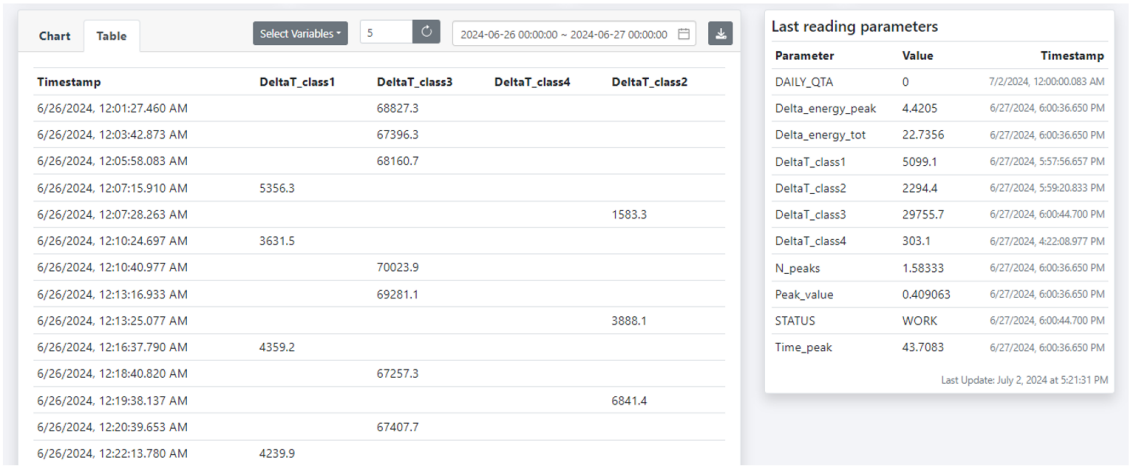

Figure 7.5: Tabular Display of Belt System Data with Parameters in a specific period

Chapter 8

Conclusions and Future Work

Due to some technical issues when deploying the trained model on the server, primarily because data is collected in real-time and factory data may not always be gathered under consistent environmental conditions, it is challenging to obtain a suitable training set for model training. This thesis only demonstrates that the LSTM-VAE model is suitable for such data. For the current TRL stage, the parameters describing data distribution on the GUI specifically the mean of the time difference data distributional are already sufficient to characterize the state of the belt system and lay a solid foundation for further product development.

8.1 Main Research Summary

This dissertation delves into the integration of data analysis and intelligent technologies to enhance the functionality and reliability of linear belt-driven motion systems within the framework of Industry 4.0. The focus of the research is on transitioning the system from TRL 4 to TRL 6, demonstrating the effectiveness of the system in both laboratory and real-world environments. Through a series of experiments and practical applications, this study validates the practicality and stability of the integrated system.

Main Contributions of This Study:

- Advanced Encoder Monitoring System: Developed an advanced encoder monitoring system capable of efficiently predicting belt wear and potential operational failures. This monitoring system employs machine learning algorithms to analyze sensor data and provide early warnings of system issues, thereby reducing unexpected downtime.
- Edge Computing Solution: Implemented an edge computing solution that optimizes data processing close to the data source. This approach not only reduces data transmission latency but also enhances the efficiency of real-time data processing, enabling the system to respond swiftly to various operational demands.
- Cloud-Based Data Analysis: Leveraged existing cloud platforms for data analysis, storing the results in the cloud, and outputting them through a remote monitoring

system. This method not only facilitates the operations of maintenance personnel but also improves the efficiency of system maintenance.

- Enhanced GUI: Enhanced the existing GUI by adding specific entries and functionalities related to this research. This improvement has streamlined maintenance and operational adjustments, making the system more user-friendly.

8.2 Impact on the Industry

The outcomes of this research have significant practical implications for the manufacturing sector, particularly within the context of Industry 4.0. By leveraging the IoT and AI, this project demonstrates how traditional belt-driven systems can evolve into intelligent, interconnected components that form part of a broader manufacturing ecosystem. Such a transformation can significantly enhance operational efficiency, reduce unplanned downtime, and enable the formulation of proactive maintenance strategies. Consequently, this leads to reduced operational costs and optimized production workflows. Specifically, the intelligent monitoring system and edge computing solutions developed in this study can assist manufacturing enterprises in achieving real-time equipment status monitoring and predictive maintenance, thereby increasing equipment utilization and production efficiency.

8.3 Limitations and Challenges

Despite the significant advancements achieved in this study, several limitations and challenges remain:

- Generality of the Model: The current LSTM-VAE model is trained on a dataset based on specific experimental conditions, which limits its applicability under different working conditions. If the system operates in varying environments, the model may fail to perform effectively.
- Cloud Platform Limitations: Although it is possible to deploy the model on cloud platforms for training, there are currently some issues with cloud platforms that make model deployment challenging. In this study, only simple data processing and analysis were conducted in the cloud, which is acceptable for the current product stage, but improvements are needed for long-term applications.
- Room for Model Performance Improvement: The performance of the existing model is not perfect and has substantial room for improvement. Particularly in handling anomaly detection and predictive maintenance, the accuracy and reliability of the model need to be further enhanced.

8.4 Future Research Directions

Future research will focus on the following areas:

- Expanding the Dataset: Increase the variety and quantity of datasets used for training, such as datasets under different conditions but with the same tension, to improve the model's generalizability and enable it to handle more diverse working environments.
- Enhancing Predictive Algorithms: Improve the predictive maintenance algorithms to increase prediction accuracy and reduce false positive rates in anomaly detection. Continuous optimization of these algorithms will enhance the system's ability to detect various potential failures.
- Data Security: Develop more robust encryption methods and secure data transmission protocols to protect sensitive industrial data. Ensuring the integrity and security of industrial data during transmission and storage is critical.
- Real-Time Optimization: Explore real-time optimization techniques that allow the system to make dynamic adjustments based on real-time data, thereby improving the system's response speed and adaptability.

In conclusion, future research will continue to advance the application of intelligent technologies within the Industry 4.0 framework, constantly enhancing the functionality and reliability of the system, and bringing more innovation and improvements to the manufacturing sector. Future research will not only focus on optimizing and improving existing technologies but also explore the integration and application of emerging technologies. Additionally, future research will aim to enhance the system's adaptive capabilities and intelligent decision-making processes, further advancing the level of intelligence in production processes. Efforts will also be made to develop a more comprehensive data security framework to ensure the safety and confidentiality of industrial data during transmission and storage. Ultimately, through multifaceted technological innovation and integrated applications, future research will provide a stronger theoretical foundation and technical support for the intelligent transformation of the manufacturing industry, driving it towards greater efficiency, flexibility, and intelligence.

Bibliography

[1] Wojciech Ptasi Jacek Kucharczyk Damian G asiorek Artur Pollak, Sebastian Temich. Prediction of belt drive faults in case of predictive maintenance in industry 4.0 platform. *Applied Sciences*, pages 1–5, 2021.

[2] Ramzan Nazim Khan Callum Webb, Joanna Sikorska and Melinda Hodkiewicz. Developing and evaluating predictive conveyor belt wear models. *Data-Centric Engineering*, pages 1–10, 2020.

[3] Guido Colombo. Cloud driven edge computing on smart systems integration. "online", 2020.

[4] D. Mazurkiewicz D.Valis1 and M. Forbelská. Modelling of a transport belt degradation using state space model. *IEEE*, pages 1–5, 2017.

[5] York Castillo Santiago Electo Eduardo Silva Lora Jose Carlos Oscar Agustin Fernando Bruno, Dovichi Filho. Evaluation of the maturity level of biomass electricity generation technologies using the technology readiness level criteria. *Elsevier*, pages 1–4, 2021.

[6] Peihua Han, Guoyuan Ellefsen, and Holmeset. Fault detection with lstm-based variational autoencoder for maritime components. *IEEE Sensors Journal*, pages 21903–21912, October 2021.

[7] Elvis Hozdic. Smart factory for industry 4.0: A review. *International Journal of Modern Manufacturing Technologies*, pages 29–31, 2015.

[8] Ryszard Blazej and Radoslaw Zimroz Jaroslaw Szrek, Jacek Wodecki. An inspection robot for belt conveyor maintenance in underground mine—infrared thermography for overheated idlers detection. *Applied Sciences*, pages 3–11, 2020.

[9] and Alois Knoll Johannes Vater, Lars Harscheidt. A reference architecture based on edge and cloud computing for smart manufacturing. *IEEE*, pages 2–5, 2019.

[10] and Welling Kingma, Diederik P. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations (ICLR)*, December 2014. arXiv preprint arXiv:1312.6114.

[11] D.V.Rama Koti Reddy P.V.S Anusha and P. Swapna. Revolutionizing conveyor belt systems: Empowering predictive maintenance with iot,cloud, and machine learning. *SSRN*, pages 2–9, 2024.

[12] Russi-Vigoya Salazar, George and M. Natalia. Technology readiness level as the foundation of human readiness level. *Ergonomics in Design*, pages 25–29, October 2021.

[13] Xiaobo Zhou Tie Qiu, Jiancheng Chi and Mohammed Atiquzzaman. Edge computing in industrial internet of things: Architecture, advances and challenges. *IEEE*, pages

2463–2469, 2020.

[14] Gabriel Lodewijksa Yusong Panga Xiangwei Liua, Daijie Hea and Jie Meib. Integrated decision making for predictive maintenance of belt conveyor systems. *Elsevier*, pages 5–7, 2019.

[15] Wolfgang Mayer Niki Patel Georg Grossmann Markus Stumptner and Ana Kuusk Zohaib Jan, Farhad Ahamed. Artificial intelligence for industry 4.0: Systematic review of applications,challenges, and opportunities. *Expert Systems With Applications*, pages 3–6, 2023.

Acknowledgements

I extend my deepest gratitude to Professors Mihai Lazarescu and Luciano Lavagno, whose exceptional guidance and mentorship have been pivotal throughout my academic journey at Politecnico di Torino. Their profound expertise, dedication to fostering an environment of academic excellence, and unwavering support have profoundly influenced my intellectual growth and professional development. The insightful feedback and encouragement I received from both professors have been instrumental in my pursuit of knowledge, inspiring me to navigate challenges with resilience and to strive for excellence in my scholarly endeavors.

I am also profoundly grateful to Mr. Alfonso Picariello, my mentor in the professional realm, whose invaluable guidance has seamlessly bridged the gap between academic theories and their practical applications in the industry. The mentorship and real-world insights provided by Mr. Picariello have enriched my understanding of the field, equipping me with essential skills and perspectives crucial for my future career. His belief in my potential and consistent support have been pivotal in my professional growth.

Furthermore, I wish to express my sincere appreciation to the esteemed faculty and staff of the Politecnico di Torino. The enriching academic environment and the diverse opportunities provided by this prestigious institution have significantly contributed to shaping my academic and professional path. The knowledge and experiences I have gained during my time here have laid a solid foundation for my future endeavors.

My journey thus far would not have been possible without the support and encouragement from these distinguished individuals and the broader academic community at Politecnico di Torino. I am indebted to them for their faith in me and for being an integral part of my educational and professional journey. I hope to repay this debt by contributing to our field with integrity, innovation, and passion.

Thank you for believing in me and for guiding me towards achieving my goals.