Steel Surface Parameterisation for On-line Quantification with Machine Learning Models

Alex Milne
[redacted]

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Doctor of Philosophy

Swansea University
Prifysgol Abertawe

Department of Computer Science
Swansea University

October 30, 2025

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ....[redacted]..... (candidate)

Date .....30/10/2025.....

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed ...[redacted]..... (candidate)

Date .....30/10/2025.....

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ..[redacted]..... (candidate)

Date .....30/10/2025.....

Abstract

Accurately estimating steel surface roughness parameters in real-time during industrial manufacturing is a critical challenge. Traditional post-production stylus-based measurements of surface roughness, specifically the arithmetic mean roughness (R_a), are slow and sample only small sections of a steel coil, leading to production inefficiencies, costly rework or downgrading of coils, and potential downstream delays in demanding just-in-time manufacturing environments due to re-production when quality is not met. While the on-line, non-contact laser reflection measurement device used by our partners offers a continuous monitoring solution, its accuracy is often insufficient compared to stylus-based ground truth measurements, negating its utility for process control.

This research explores the application of machine learning (ML) to enhance the accuracy of surface roughness estimation from raw laser reflection data. By leveraging both limited labelled and abundant unlabelled industrial datasets, this work investigates various ML-driven methods. These include: data imputation techniques aimed at improving existing calculation methods by addressing data quality issues; end-to-end models designed for direct mapping from sensor input to the stylus-equivalent R_a value; efficiency enhancements for promising model architectures; and the incorporation of representation learning via self-supervised learning (SSL) to effectively utilise unlabelled data and overcome labelled data scarcity.

The findings demonstrate the substantial advantages of ML approaches over conventional methods. End-to-end deep learning models consistently achieve significantly higher accuracy in predicting R_a compared to the baseline sensor calculations. Our comprehensive comparative studies reveal that architectures such as Temporal Convolutional Networks (TCNs) and certain 2D Convolutional Neural Networks provide a robust combination of high predictive accuracy and computational feasibility for this specific industrial data type. Furthermore, leveraging large unlabelled datasets via pretraining proved highly effective; representation learning strategies, particularly a novel task-specific coil classification pretext task and a TCN-based autoencoder, yielded models with enhanced generalisation and accuracy upon fine-tuning with

limited labelled data. Methodological insights also include the necessity of a statistics-aware loss function for imputation tasks requiring the preservation of surface texture characteristics, and the successful GPU acceleration of the ROCKET model for enhanced throughput.

The key contributions of this thesis therefore encompass not only the successful application and comparison of various ML techniques (imputation; end-to-end regression; and pretraining strategies like contrastive learning, autoencoding, and coil classification) to this specific industrial problem, but also methodological advancements such as the statistics-aware imputation loss and efficiency improvements to the ROCKET model.

These findings show that machine learning models, particularly those enhanced with representation learning techniques, can significantly improve the accuracy of on-line surface roughness estimation. This paves the way for reliable real-time quality control, enabling faster process feedback, reducing waste, facilitating process optimisation, and ultimately opening possibilities for automated closed-loop control systems in steel manufacturing.

Acknowledgements

This thesis would not have been possible without the support, guidance, and encouragement of many people, to whom I am sincerely grateful. First, I would like to express my deepest thanks to my primary supervisor, Prof. Xianghua Xie, for his guidance, support, and expertise throughout the course of my PhD. I am also grateful to my second supervisor, Dr. Gary Tam, for his insights and feedback during this process. I am especially thankful for the collaboration with Tata Steel, whose data formed the basis of much of this work. In particular, I would like to thank Pieter Baart (Tata Steel Netherlands), and Sam Gilroy, Steve Jones, and Christopher Hague (Tata Steel Port Talbot) for their time, input, and support. I would also like to thank everyone in the Vision Group at Swansea University for the engaging discussions and shared meetings, which were a valuable part of my research environment. During the course of my PhD, I had the opportunity to undertake an internship at Expedia Group. This experience helped me develop not only technical insights into industrial AI applications, but also essential professional skills. I also participated in an Alan Turing Institute Data Study Group, which was a rewarding experience in applying AI collaboratively to real-world challenges faced by industry and partners. I owe a great deal to those who supported me personally throughout this journey. My heartfelt thanks go to Sophie Sadler, for her constant support and patience over the years. I am also grateful to the Swansea University Mountaineering Club and the Snowriders for the friendships, trips, and much-needed escapes which provided a welcome balance to academic life. Finally, I would like to thank my parents for their unwavering support throughout my education. This thesis is dedicated to them.

Contents

List of Publications

The following is a list of publications arising from the works in this thesis.

1. A. Milne and X. Xie, “Steel surface roughness parameter prediction from laser reflection data using machine learning models,” *The International Journal of Advanced Manufacturing Technology*, 2024. [85]

2. A. Milne, X. Xie, and G. K. L. Tam, “Pretraining techniques for Ra prediction with long thin spatial industrial data,” in *Proceedings of Advanced Concepts for Intelligent Vision Systems (ACIVS)*, 2025, accepted for presentation. [86]

List of Acronyms

AI	Artificial Intelligence
BOS	Basic Oxygen Steelmaking
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DL	Deep Learning
DTW	Dynamic Time Warping
EAF	Electric Arc Furnace
EDT	Electric Discharge Texturing
FCN	Fully Convolutional Network
FFT	Fast Fourier Transform
GA	Galvannealed
GAP	Global Average Pooling
GI	Galvanised
GPU	Graphics Processing Unit
ISO	International Organization for Standardization
KL	Kullback–Leibler divergence
LSTM	Long Short-Term Memory
ML	Machine Learning
MSE	Mean Squared Error
MZ	MagiZinc
PPV	Proportion of Positive Values
RNN	Recurrent Neural Network
RMSE	Root Mean Squared Error
ROCKET	RandOm Convolutional KErnel Transform
RPc	Peak Count (surface parameter)
Rq	Root Mean Square Roughness
Rsk	Skewness (surface parameter)
Rz	Maximum Height of the Profile
SAITS	Self-Attention-based Imputation for Time Series
SSL	Self-Supervised Learning

TCN	Temporal Convolutional Network
TSER	Time Series Extrinsic Regression
TSC	Time Series Classification
UCR	UCR Time Series Archive
VAE	Variational Autoencoder
ViT	Vision Transformer

List of Tables

List of Figures

Chapter 1

Introduction

Modern high-volume manufacturing across many industrial sectors operates under the dual pressures of maximising production speed and ensuring exceptionally high product quality. Complex, often high-speed processes occurring in demanding environments mean that even subtle variations can significantly impact the final product's adherence to stringent specifications. Any quality control method must provide speed, responsiveness, fast actionability, and coverage required for comprehensive process monitoring; failing this can lead to potential inefficiencies, undetected flaws, and costly waste. Consequently, there has been a widespread shift towards integrating advanced production sensor systems that generate vast quantities of real-time data. However, extracting meaningful, actionable insights from this often noisy, high-dimensional, and rapidly changing data stream poses a significant analytical challenge, frequently overwhelming conventional mathematical models derived from physical principles or simple rule-based alert systems. This gap, the need for robust methods to rapidly translate complex sensor data into reliable quality assessments for real-time process control, is a key area where Artificial Intelligence (AI) and Machine Learning (ML) techniques offer substantial promise.

This thesis applies such advanced machine learning methodologies to address a critical quality control bottleneck within a specific foundational industry: steel manufacturing. In this domain, achieving precise surface characteristics is paramount, particularly for high-value products destined for sectors such as automotive where texture directly impacts downstream performance in processes such as stamping and coating. Specifically, we focus on the challenge of accurately quantifying micro-scale surface roughness during the temper rolling process.

While non-contact optical sensors offer the potential for continuous, real-time monitoring, their current implementations often lack the required accuracy compared to established stylus-based methods, primarily due to the complexities of light-surface interactions and process variability. The core aim of this research is therefore to investigate and develop machine learning models capable of bridging this accuracy gap, transforming raw optical sensor data into reliable, real-time surface roughness predictions specifically for steel production.

1.1 Steel Surface Roughness: Challenges and Motivation

Steel is a fundamental material in modern industry, playing a crucial role in sectors such as automotive, construction, and infrastructure. Achieving precise surface characteristics is a key requirement in high-value steel products, particularly for automotive paneling, where surface texture affects press performance and paint adhesion [11,39]. One of the most critical processes for refining steel surfaces is temper rolling, in which the steel strip is cold-rolled to improve its mechanical and surface properties.

The temper rolling process involves a series of adjustable parameters—including roll selection, roll force, and speed—which influence the final surface texture. The rolls themselves have a predefined surface texture, imparted by electric discharged texturing (EDT). The rolls then transfer this texture onto the steel under specific pressure and speed conditions. The adjustable parameters allow line operators to tailor the steel's roughness to meet customer specifications. However, traditional measurement feedback on whether the desired surface texture has been achieved is very accurate but is extremely slow and inefficient.

Currently, surface roughness is measured post-production using a stylus-based method, which only samples small sections of a steel coil, typically at the head or tail [25]. This approach presents two major issues: (1) these small samples may not be representative of the entire coil, and (2) feedback is delayed, preventing mid-coil adjustments that could improve production efficiency. If the steel does not meet customer requirements, the coil's value is reduced, potentially necessitating multiple re-production cycles, which is costly and time-consuming.

In just-in-time manufacturing environments [107], such inefficiencies can lead to downstream delays and even production stoppages for customers awaiting replacement materials. To address this, on-line surface measurement techniques have been developed [26], allowing for continuous, non-contact monitoring using laser reflection sensors. One example works by firing a laser at the surface, capturing scattered light intensities at various angles. By analysing

these reflections, surface gradients can be calculated, and roughness parameters such as R_a, the mean deviation of the roughness profile, can theoretically be inferred.

However, these optical measurement techniques can lack sufficient accuracy when compared to stylus-based ground truth measurements. The complex physical processes influencing non-contact height measurements introduce unknown factors, making precise closed-form calculations of R_a inaccurate and challenging.

Direct comparisons between the stylus and on-line results can reveal discrepancies between the optical system's output and the industry-standard stylus results, particularly across diverse surface types and under real-world production conditions. While the underlying physical principles are complex, the observed performance gap means these systems are often not trusted for final quality assurance or precise process control.

This lack of sufficient, verifiable accuracy significantly limits the practical utility of these on-line systems. If the real-time feedback is unreliable, operators cannot confidently make process adjustments, potentially hindering rather than helping quality control efforts. Furthermore, the absence of trusted, high-fidelity on-line measurements prevents the development and implementation of sophisticated automated closed-loop control systems for processes like temper rolling. This represents a major barrier to maximising production efficiency, minimising costly waste from rework or downgraded coils, and ensuring consistent adherence to demanding customer specifications, especially in responsive manufacturing environments like just-in-time production. Bridging this accuracy gap between on-line sensing and the established ground truth is therefore a critical step towards unlocking the full potential of real-time quality control and automation in steel manufacturing.

1.2 Machine Learning for Improved Surface Roughness Estimation

Artificial intelligence (AI) and machine learning (ML) have transformed industries by enabling automated decision-making, predictive modelling, pattern recognition, and enable deeper insights into complex data across vast datasets. ML models excel at identifying complex relationships within data that traditional rule-based systems struggle to capture. These models learn patterns from examples rather than relying on predefined equations, making them highly effective for tasks where precise physical modelling is difficult or impractical. Applications of ML span across computer vision, natural language processing, financial forecasting, and industrial

automation, with rapid advancements in deep learning further enhancing capabilities.

In manufacturing, ML plays a crucial role in process optimisation and quality control, particularly in environments where large amounts of sensor data are collected. One such challenge is the issue central to this thesis: accurate surface roughness estimation in steel production. While traditional measurement techniques like stylus profilometry are slow and limited, laser-based optical sensing, though fast, suffers from inaccuracies. This is where machine learning offers significant potential.

To improve on-line roughness estimation and address the accuracy gap, this thesis explores machine learning-based methods that improve the flow from the laser reflection data into accurate surface roughness predictions. Rather than relying solely on traditional physical models, machine learning can account for unknown variations in material properties, measurement conditions, and signal distortions. We use two distinct and competing machine learning philosophies for this purpose.

The first philosophy is to treat the baseline's limitations as a data quality problem. The raw laser signal suffers from noise and signal dropouts. Therefore, one approach is to apply machine learning to "clean" this signal before it is processed. This involves developing models to identify and impute these compromised data segments, with the hypothesis that a complete, high-fidelity signal will yield a much more accurate R_a value from the existing closed-form solution.

The second, more direct, philosophy is to bypass the closed-form solution entirely and replace it with an end-to-end model. This approach assumes the physical model itself is the limiting factor or that identifying and imputing the poor regions is more difficult. We therefore train various ML models to learn the complex, non-linear mapping directly from the raw, noisy laser reflection data to the final stylus-derived R_a values. As part of this investigation, we also develop and evaluate enhancements to promising model architectures, such as the ROCKET model, to optimise them for this specific industrial task.

A critical challenge for either approach is the scarcity of high-quality labelled data. As detailed in Chapter 4, obtaining stylus-based ground truth labels is a costly and time-consuming lab process. By contrast, laser-based measurements taken on-line allow for the collection of vast quantities of unlabelled data directly without corresponding stylus measurements. This disparity motivates another major contribution of this thesis: leveraging representation learning and self-supervised learning (SSL) to improve model performance.

These pretraining methods are powerful in domains where acquiring labelled data is infea-

sible [47], as is the case in steel production [70,105]. Although extensively studied in computer vision [19,50] and time-series analysis [81,127], their application to industrial surface metrology remains relatively underexplored. This thesis investigates several pretraining strategies, such as contrastive learning, autoencoding, and a coil classification pretext task, to learn robust data representations from the large unlabelled dataset. These representations are then fine-tuned on the small labelled dataset, with the goal of building a final predictive model that is more accurate and generalises more effectively than a model trained on the limited labelled data alone.

1.3 Overview of Contributions

This work contributes to the advancement of AI-driven manufacturing by demonstrating how machine learning can enhance industrial surface monitoring. Note that more specific contributions and research questions are laid out in Chapter 5.

Primarily, we demonstrate that end-to-end deep learning models can significantly enhance the accuracy of predicting the critical R_a parameter directly from raw laser sensor data, substantially outperforming existing mathematical methods derived from physical properties (detailed in Chapter 8).

We conduct a comprehensive comparative study identifying specific architectures, such as Temporal Convolutional Networks and 2D Convolutional Networks, that offer a strong balance of accuracy and computational efficiency suitable for this industrial context. Within this study, we also contribute specific enhancements to the ROCKET model [28], introducing implementation changes for improved speed and efficiency, experimentally assessing their impact, and exploring the potential for backpropagation-based kernel training (Chapter 7).

Recognising the limitations of labelled data availability, we further make significant contributions by exploring representation learning. We introduce and evaluate multiple pretraining strategies, including contrastive learning, a novel TCN-based autoencoder, and a task-specific coil classification pretext task. We provide a comparative study of these supervised vs. self-supervised approaches, highlighting the trade-offs and potential challenges in industrial model deployment. This demonstrates that leveraging large unlabelled datasets can further boost predictive performance, offering insights applicable to other industrial pretraining scenarios (Chapter 9).

Alternatively, addressing the limitations of mathematical method derived from physical properties as a data quality issue, our efforts explore the application of machine learning to the

imputation of compromised segments, aiming to improve the accuracy of the original closed-form solution (Chapter 6).

Through this research, we aim to bridge the gap between optical sensing techniques and traditional surface roughness evaluation, ultimately enabling real-time quality control, reducing waste, and improving overall production efficiency in steel manufacturing.

The remainder of the thesis is structured logically to build the context necessary for understanding the data-driven contributions. First, Chapter 2 provides essential background on the steelmaking industry, with particular focus on processes which apply our research such as surface metrology, and fundamental machine learning concepts. This is followed by Chapter 3, which critically reviews related work in ML for manufacturing, time series analysis, and representation learning, identifying key research gaps.

Crucially, the specific challenges and novelty of this research are deeply intertwined with the unique characteristics of the industrial dataset and the processes which capture it. Therefore, Chapter 4 details the data acquisition process, the structure of the labelled and unlabelled datasets, the existing baseline calculation, and inherent data challenges such as spatial misalignment and noise.

This chapter order presents the data (Chapter 4) before the formal problem statement (Chapter 5). This differs slightly from the typical structure of placing the problem statement immediately after the related work, but the order is chosen deliberately. Fully appreciating the specific research questions and the novelty of the approaches developed requires understanding not only the literature gaps (Chapter 3) but also the unique characteristics, limitations (such as the many-to-many labelling problem detailed in Section 4.3.1.1), and opportunities presented by the industrial dataset detailed in Chapter 4. Presenting the data first provides this essential context.

With the data context established, Chapter 5, the Problem Statement Chapter, summarises the literature gaps and data challenges, transforming these into specific research questions, as well as presenting our hypotheses and a summary of contributions that guide the technical work.

Subsequent chapters contain the key methodological and experimental contributions of this thesis. Chapter 6 addresses data quality, introducing a methodology for identifying and imputing missing data gaps in surface profile data, before performing a comparative study of various approaches to impute them. In this study, we compare deep learning approaches with basic interpolation methods. This approach has the potential to improve the accuracy of the

closed-form solution calculations.

However, it is also possible to bypass the use of a closed-form calculation entirely by using machine learning as an end-to-end approach for predicting the steel surface roughness. In preliminary experiments, we found that one promising approach for making predictions was the ROCKET model, however there were opportunities to further improve both its efficiency and accuracy. Therefore, in Chapter 7, we explore enhancements to this model architecture, introducing a GPU-accelerated implementation of the model, and second through evaluations of various pruning and fine-tuning approaches.

Having established improvements, we then perform a full comparative study of a wide variety of supervised models for directly predicting surface roughness in Chapter 8. These include both 1D and 2D deep learning approaches, in addition to non-deep learning models such as ROCKET. We further build on this supervised evaluation in Chapter 9, where we investigate the effect of pretraining on large unlabelled datasets. We investigate various approaches for pretraining, including both generative and contrastive methods, and a coil classification task designed specifically for our dataset.

Finally, concluding remarks are made in Chapter 10, where we review our research questions and hypotheses in light of the findings as well as summarising avenues for future work.

Chapter 2

Background

In this chapter, we introduce relevant context to support the reader in understanding the content of this thesis. Throughout the thesis, our focus is on the application of machine learning methods to predictive tasks found within the steel manufacturing industry. Therefore, we first provide relevant background on both the steel industry and machine learning models. We begin by presenting a comprehensive background on surface metrology in steel manufacturing in section 2.1, in order to provide a high-level introduction to the domain in which our contributions are made. We then expand on some foundational concepts in surface analysis in section 2.2 which are particularly pertinent to the predictive tasks we aim to solve. Following this, we then introduce the fundamental concepts of machine learning in section 2.3 as well as specific model architectures well-suited to our datasets in section 2.5.

The goal of presenting this background is to arm the reader with the prerequisites to understand the work of this thesis. Separately, we then explore related work in the next chapter, Chapter 3, where our aim is to outline the existing work which closely relates to ours in order to establish which research directions have already been explored, and how our approaches differ to those previously introduced.

2.1 Industrial Context: Surface Metrology in Steel Manufacturing

While the core contributions of this thesis are in the domain of machine learning, they are motivated by and applied to a specific industrial challenge. Therefore, a foundational understanding of the steel manufacturing process is essential to appreciate the context and significance

of the problem. While this thesis focuses specifically on surface metrology, achieving consistent quality across various parameters (e.g., mechanical properties, chemical composition) is a pervasive challenge throughout the steel manufacturing process, addressed through various monitoring and control strategies.

The traditional steelmaking process begins with raw materials such as iron ore [91], which is crushed and sintered, creating porous lumps to prepare it for the next steps. Coal is also crushed and heated up to create a hot burning high-carbon derivative known as coke. The sintered iron is combined with coke in a blast furnace at exceptionally high temperatures of around $900 - 1200\,^{\circ}\mathrm{C}$ [82] to produce an intermediary product known as pig iron. This pig iron has a high carbon content, which must be reduced to create useful steel. Historically, this was achieved with methods like the Bessemer process (c. 1850s), which enabled the first mass production of steel, and later the Open-Hearth furnace (c. 1880s), which allowed for better quality control over longer heating periods.

Modern steelmaking predominantly uses one of two methods: Basic Oxygen Steelmaking (BOS) or the Electric Arc Furnace (EAF). For many years, BOS has been the more common method for large-scale production, where high-purity oxygen is blasted through the molten iron to reduce carbon content rapidly, a process that reduced the steelmaking cycle from over 8 hours in an open-hearth furnace to around 45 minutes [109]. BOS has remained dominant, while EAF is typically used for specialised or recycled steel production, partly because of the higher electricity requirements. Nevertheless it remains important due to its allowance of precise quality control via slag absorption [75]. The EAF method primarily uses scrap steel as its input, which can be sourced from old cars, construction materials, and other recycled goods. This highlights a key property of steel: it is 100% recyclable and can be endlessly reused without loss of quality [91].

2.1.1 Our Industrial Partner and Process

Throughout this thesis, we collaborated with an industrial partner, Tata Steel. The project was initially a partnership with the joint Tata Steel Europe R&D division, with major buy-in from members in the Netherlands. However, following a corporate restructuring which separated the UK and Dutch arms of the company, the project's industrial supervision was transferred to the facilities in South Wales [54].

The Port Talbot plant has historically operated as an integrated steelworks, and implemented BOS converters in 1969, when the open-hearth furnaces were replaced as part of a

modernisation effort [44]. As of 2024, the blast furnaces are being decommissioned as part of a strategic pivot towards a more sustainable Electric Arc Furnace-based production model [116]. Throughout its history, the facility has produced hot-rolled coils of flat-rolled sheet steel.

Once the production steel has been hot-rolled and coiled, it is transported from Port Talbot to the Llanwern Works for finishing. Here, the steel first passes through a Pickling Line, which is a series of acid baths that remove any surface rust or scale that has formed. After pickling, the steel undergoes several cold rolling processes, which prepares the final steel products to be sold to customers. In this process, a galvanised coating can be applied via the ZODIAC line, which produces both standard zinc (GI) and MagiZinc (MZ) coatings to prevent corrosion. In the final steps of the cold rolling line, in the process central to this thesis, the critical final stage is Temper Rolling, which is performed to impart the final mechanical and surface properties to the flat-rolled sheet steel before coiling and delivery.

2.1.2 Temper Rolling

Temper rolling is a critical final stage in strip steel production in the cold rolling process. It involves a final cold-rolling of the steel to improve its mechanical properties and, crucially, to control its surface texture. For high-value products like automotive panelling, specific surface properties are contractually required to ensure good press performance in stamping operations [39] and high-quality paint adhesion [11]. This desired surface texture is imparted onto the steel sheet by large work rolls, which are themselves textured using a process known as Electric Discharge Texturing (EDT). During production, line operators can adjust key process parameters, such as roll force and strip elongation, to meet customer specifications.

The primary challenge in this process is the slow feedback loop. Surface measurements are traditionally taken post-production using a stylus device on a small section of the coil taken from the ends. This can lead to two main issues: the samples may not be representative of the entire coil surface [25], and the delayed feedback does not allow for real-time process adjustments. This can result in entire coils failing to meet customer requirements, leading to reduced value and costly re-production.

2.1.3 On-line Measurement Methods

To address the slow feedback loop of manual stylus measurements, various non-contact optical systems have been developed [94]. The evolution of these systems began with lab-based prototypes, such as the work by Luk et al. [79], which used a microscope and gray-level his-

tograms to estimate roughness from scattered light on stationary samples. This foundational work paved the way for more robust, commercially available on-line systems designed for active production lines, such as those detailed by Bilstein et al. [12]. The system used in this thesis, the **SORM 3plus** by EMG, operates on a similar principle of laser-light scattering.

However, as noted in the introduction and detailed later, translating the complex signals from such optical systems into consistently accurate roughness parameters using purely physics-derived calculations remains challenging, motivating the exploration of data-driven machine learning approaches investigated in this thesis.

The SORM 3plus, a commercial on-line measurement device manufactured by EMG, serves as the foundation of the data acquisition process. The device, whose measurement head is illustrated in Figure 2.1, is designed to non-destructively assess the steel surface during production. It consists of a single laser emitter and a semicircular array of 20 discrete photosensors. The laser is positioned precisely between the central two sensors in the array, and the sensors are arranged to capture the angular distribution of reflected light.

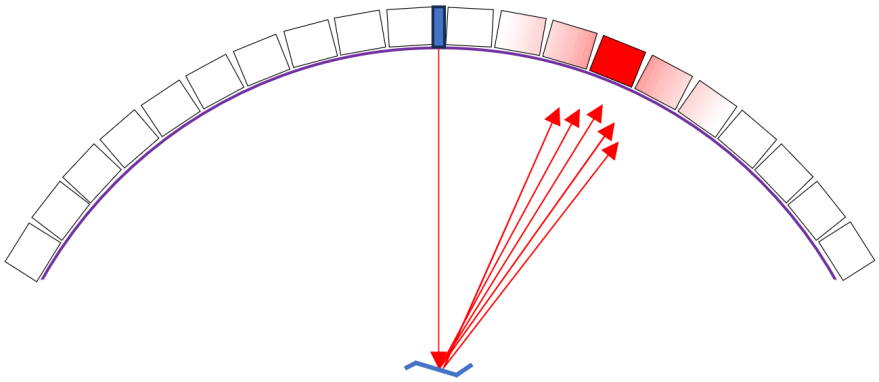


Figure 2.1: An illustrative diagram of the measurement head of the on-line laser measurement device. A laser is directed at the steel surface, and the reflected light is captured by a 1D array of 20 sensors. The intensities recorded by these sensors at a single time step form a 20×1 vector in the final dataset.

The sensors are spaced at 6.7-degree increments, spanning a total arc of 127.4 degrees, with a 6.8-degree gap at the centre to accommodate the laser emitter. To perform a measurement, a laser is directed at the steel surface. The reflected light is captured by the sensor array, producing a 1D "image" of light intensity versus reflection angle, where each of the 20 sensors acts as a pixel in the image. This measurement is repeated at discrete time intervals, generating many such 1D images.

As the steel strip moves along the production line, the device remains in a fixed position, scanning the surface along the rolling direction. The measurement pulse rate is synchronised

with the line speed, ensuring that each measurement is taken at a constant spatial interval. This process yields a sequence of intensity-angle measurements over distance, which can be represented as a 2D data matrix of size $\mathbb{N}^{C\times T}$, where $C = 20$ is the number of sensors and T is the number of measurement steps. The intensity values from the sensors are digitised as 8-bit integers, ranging from 0 to 255. The SORM 3plus device uses this data to internally calculate surface parameters of the surface texture.

Although the recorded data are two-dimensional (sensor intensities × time), the physical measurement itself follows a one-dimensional track. The laser illuminates and tracks a single line along the steel surface as the strip moves through the measurement head, and no information is captured across the transverse direction of the strip. The 2D data therefore represent angular light reflections collected over time from a single 1D surface track, rather than a two-dimensional spatial scan of the steel surface. Consequently, surface roughness parameters such as R_a are derived from line-based surface information only, not from full areal surface measurements.

2.2 Foundational Concepts in Surface Analysis

2.2.1 The Roughness Profile

The basis for calculating surface statistics is the **roughness profile**. This is derived from a raw surface height profile (such as one from a stylus) by filtering out long-wavelength components, classified as 'waviness'. This separation is governed by international standards. Specifically, **ISO 16610-21** [2] defines the mathematical properties of the standard Gaussian profile filter, while **ISO 4288:1996** [1] is the procedural standard that specifies which filter cutoff wavelength (λ_c) to use. The selection of λ_c is critical and is based on the type of surface being measured. For instance, ISO 4288:1996 recommends specific cutoff wavelengths based on the expected range of the R_a value for non-periodic profiles; for surfaces typically encountered in this work (e.g., R_a between 0.1 µm and 2.0 µm), a cutoff of 0.8 mm is specified. Applying this filter results in a low-frequency waviness profile. Subtracting this waviness profile from the original signal results in the roughness profile, which represents the high-frequency texture that is the focus of this work. A full technical implementation of an approximation of this filtering process is detailed in Section 4.2 of Chapter 4.

2.2.2 Key Surface Statistics

The primary metric used to characterise surface texture in this work is the **arithmetic mean roughness (R_a)**. It is defined as the arithmetic mean of the absolute values of the profile's deviations from its mean line over an evaluation length L:

$$R_a = \frac{1}{L}\int_0^L |z(x)|dx \approx \frac{1}{N}\sum_{i=1}^{N}|z_i| \tag{2.1}$$

where $z(x)$ (or z_i) is the height of the roughness profile relative to its mean line. R_a is widely regarded in industry as the primary contractual parameter for sheet steel products [5, 38, 117] and is therefore the sole focus of the predictive tasks in this thesis.

While R_a is the most prevalent metric and the only one used in this thesis, other parameters are also used to characterise different aspects of a surface [94]. Other common statistics include:

- **Root Mean Square Roughness (R_q):** The root mean square of the profile height deviations. It is more sensitive to large peaks and valleys than R_a and is defined as:

$$R_q = \sqrt{\frac{1}{L}\int_0^L z(x)^2 dx} \approx \sqrt{\frac{1}{N}\sum_{i=1}^{N} z_i^2} \tag{2.2}$$

- **Maximum Height of the Profile (R_z):** It is useful for identifying the presence of extreme, potentially problematic surface features. It represents the vertical distance between R_p, the highest peak height above the mean line, and R_v, the maximum valley depth below the mean line of the roughness profile within the evaluation length.

$$R_z = R_p + |R_v| \tag{2.3}$$

  Since R_v is defined as a negative quantity, the absolute value ensures that R_z represents the total vertical distance between the highest peak and lowest valley of the profile. This definition corresponds to the profile-based maximum height parameter.

- **Skewness (R_{sk}):** Measures the asymmetry of the height distribution. A negative skew indicates a surface dominated by valleys, while a positive skew indicates a surface of peaks. Given the height profile $z(x)$ over the sampling length L, with mean height $\bar{z}$, the skewness is the normalised third moment about the mean:

$$R_{sk} = \frac{1}{R_q^3}\frac{1}{L}\int_0^L (z(x)-\bar{z})^3\, dx \tag{2.4}$$

However, this must be approximated when calculated sampled data, using the following discrete formula:

$$R_{sk} = \frac{1}{nR_q^3} \sum_{i=1}^{n} (z_i - \bar{z})^3 \tag{2.5}$$

where $z_i, i = 1, \ldots n$ are equally spaced points with mean height $\bar{z}$.

- **Peak Count (R_{Pc}):** A standardised count of the number of local peaks in a profile, per unit length (typically 10mm or 1cm). This can be useful in settings such as sealing (where peak density affects leak tightness), coatings and paint appearance.

$$R_{Pc} = \frac{N}{L} \times L_{\text{standard}} \tag{2.6}$$

where N is the number of qualifying peaks within the evaluation length, L is the evaluation length, and L_{standard} is the standard normalisation length (e.g., 10 mm).

Although this thesis focuses exclusively on predicting R_a, the machine learning frameworks developed could be readily adapted to predict these (or other parameters) as an alternative or at the same time, for example using multivariate regression. This is left as future work, but is a key direction that follows directly from the results of this thesis.

2.3 Fundamental Concepts in Machine Learning

At its core, machine learning (ML) is a subfield of artificial intelligence focused on building systems that learn patterns directly from data, rather than being explicitly programmed with rules. A model is typically developed with the goal to learn a function that can accurately map new, unseen inputs to their correct outputs. The dominant paradigm within ML is the use of **neural networks**, which are powerful function approximators. **Deep learning** refers to the use of neural networks with many layers, which can learn hierarchical representations of complex data. These approaches mark a paradigm shift from traditional software, where logic is hard-coded by a programmer. This shift towards learning from data is particularly relevant for complex industrial problems like surface roughness estimation from indirect sensor readings, where precisely modelling all physical interactions and noise sources is often inaccurate.

Initially, such problems might be approached with a **rule-based system** or a physics-based **closed-form solution**, like the one detailed in Section 4.2. This requires significant domain expertise to manually define a fixed, mathematical model that transforms inputs to outputs.

The second approach is that of **classic machine learning**, which introduced the ability to learn from data, but still relied heavily on human expertise. In this paradigm, a domain expert performs **manual feature engineering**, carefully designing a process to extract meaningful features from the raw data (e.g., statistical moments, frequency components). A relatively simple model, such as a linear regressor, is then trained to map these handcrafted features to the final output. This reliance on manual feature design is not only time-consuming but can also introduce human bias, potentially limiting model performance. The ROCKET architecture, discussed later, shares philosophical similarities with this approach, using a fixed (albeit random) transformation to generate a large bank of features instead of manual feature engineering.

The third paradigm is that of **end-to-end learning** using **neural networks**, which remove the need for manual feature engineering. While the concept of neural networks as universal function approximators has existed for decades, their potential was limited to shallow architectures. The advent of **deep learning** was enabled by several key breakthroughs. The backpropagation algorithm [102] provided an efficient method for training deep networks, which had not been needed for models of the second paradigm. Non-saturating activation functions like the Rectified Linear Unit (ReLU) [69], first introduced in the paper by Krizhevsky et al. which proposed the AlexNet neural network architecture, mitigated the vanishing gradient problem. Meanwhile, architectural innovations like skip connections [51] allowed gradients to flow through extremely deep networks. These advances allow models like Convolutional Neural Networks (CNNs) to learn the entire processing pipeline, from raw data to final output, including the optimal hierarchical features, automatically. This ability to learn complex representations directly from raw sensor data is a central premise of the predictive models developed in this thesis.

2.3.1 Supervised Learning

The primary task in this thesis is to achieve a model which can help us produce an R_a value from the input laser data. One method could be **supervised learning**, specifically regression. Formally, we are given a dataset D consisting of N input-output pairs, $D = \{(X_1, y_1), (X_2, y_2), ..., (X_N, y_N)\}$. Each input X_i is a high-dimensional laser reflection profile, and each corresponding target y_i is a scalar ground-truth roughness value, R_a. The goal is to learn a function, or model, f, parameterised by a set of weights θ, that can accurately map any new input X to its corresponding output y:

$$\hat{y} = f(X; \theta) \tag{2.7}$$

where $\hat{y}$ is the model's prediction of the true value y. The process of "learning" involves finding the optimal parameters θ that minimise the prediction error across the entire training dataset, as measured by a loss function.

Deep learning models mainly consist of fully connected, convolutional, Recurrent Neural Network (RNN), and transformer "layers", each of which is composed of artificial "neurons" which perform complex transformations. These layer types differ as follows. Fully connected networks are the most basic type, where neurons of one layer are all connected to all the neurons of both the previous and next layer. This results in a matrix multiplication followed by a non-linear activation function. Meanwhile, convolutional layers use parameter sharing and multiply a filter of weights (also known as a kernel) over the input, using a sliding window. By contrast, RNNs have cyclic connections where a deeper layer connects to a previous layer. Finally, transformers use the self-attention mechanism, which utilises fully connected layers to generate a query, a key, and a value. These are then used to learn and subsequently provide relationships between different points in the input sequence.

2.3.2 Common Evaluation Metrics

To evaluate model performance throughout this thesis, we use standard, commonly-used metrics for both regression and classification tasks. We first introduce some of the most pertinent metrics for regression tasks:

- **Mean Squared Error (MSE):** The average of the squared differences between the predicted and actual values. It heavily penalises large errors.

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{2.8}$$

- **Root Mean Squared Error (RMSE):** The square root of the MSE. This is often preferred as it returns the error in the same units as the target variable (e.g., in μm).

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{2.9}$$

- **Pearson's Correlation Coefficient (r):** Measures the linear correlation between the predicted and actual values. It ranges from -1 to +1, where +1 indicates a perfect positive linear correlation.

Classification: For classification tasks, such as in the benchmarks on the UCR archive, performance is measured by:

- **Accuracy:** The proportion of correctly classified samples. It is the most common metric but can be misleading on imbalanced datasets.
- **Precision, Recall, and F1-Score:** These metrics are useful for imbalanced classification. **Precision** is the proportion of positive identifications that was actually correct. **Recall** is the proportion of actual positives that was identified correctly. The **F1-Score** is the harmonic mean of precision and recall.

2.3.3 Time Series Analysis

A time series is a sequence of data points $X = (x_1, x_2, ..., x_T)$ indexed in chronological order $t = 1, ..., T$. In the context of this thesis, a single laser scan can be treated as an image where the 20 sensor channels become pixels or a **multivariate time series**, where each $x_t \in \mathbb{R}^{20}$ is a vector of the 20 sensor readings treated as channels at a given timestep. Traditionally, time series models like Recurrent Neural Networks (RNNs) process this data sequentially, maintaining a "state" that is updated at each timestep based on the previous one. This approach explicitly models the temporal dependency of one point on the next.

However, many modern deep learning architectures, such as the Convolutional Neural Networks (CNNs) and Transformers used in this work, adopt a different perspective. They treat the entire time series as a single, holistic input. Instead of processing point-by-point, they use mechanisms like convolutional filters or self-attention to learn patterns and relationships across the entire sequence simultaneously. This is particularly effective for our data, where identifying global spatial patterns across the entire scan is often more important than modelling the precise step-by-step temporal evolution.

A related supervised learning task explored in this thesis is imputation. Given a data series with missing or corrupted values, the goal is to train a model that can predict these missing values based on the observed context. This can be framed as a regression problem where the model learns to predict a subset of the data points from the others. Suppose a data vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ contains missing indices at a subset of indices $\mathcal{M} \subset \{1, 2, \ldots, n\}$, while the remaining indices are observed data $\mathcal{O} = \{1, 2, \ldots, n\} \backslash \mathcal{M}$. The imputation problem is to estimate the missing values $\{x_j : j \in \mathcal{M}\}$ using the observed values $\{x_i : i \in \mathcal{O}\}$. Often, a

regression-based imputation model is trained to minimise the prediction error:

$$\min_{\theta} \sum_{j \in \mathcal{M}} \mathcal{L}\left(x_j, f_\theta(\{x_i : i \in \mathcal{O}\})\right) \tag{2.10}$$

where f_θ is the regression function parameterised by θ (for example, a neural network with weights θ), and $\mathcal{L}$ is a loss function such as mean squared error.

2.4 Representation and Self-Supervised Learning

Representation learning is a field of machine learning focused on automatically discovering optimal features or "representations" of data for a given task. Instead of relying on handcrafted features, the goal is to learn a data transformation $g(X)$ that maps raw data X into a new vector space where key properties are more salient and the task (e.g., classification or regression) becomes easier to solve. For a neural network, the entire model up to the final layer can be seen as a representation learner. Autoencoders [9] and variational autoencoders (VAEs) [66] are two classic examples of unsupervised representation learning models. An autoencoder is trained to reconstruct its own input, forcing its internal layers to learn a compressed, salient representation. Meanwhile, VAEs extend the autoencoder framework with a probabilistic perspective. Instead of encoding the input as a fixed vector, VAEs learn a distribution over the latent space, typically parameterised as a Gaussian with mean and variance output from the encoder.

Self-Supervised Learning (SSL) is a powerful family of approaches for learning representations from (typically large) unlabelled datasets. It bridges the gap between unsupervised learning (no labels) and supervised learning (provided labels). The core idea of SSL is to create a **pretext task**, essentially mimicking supervised learning but in a setting where the supervision signal is derived from the data itself. Some families of SSL include:

- **Generative methods**, which use models such as the aforementioned autoencoders, learn representations by trying to generate or reconstruct the data. For example, a model might be trained to predict a missing part of an input.
- **Contrastive methods** models are trained to determine if two augmented versions of an input came from the same source. E.g. SimCLR [20], which learn representations by pulling augmented views of the same sample ("positives") closer together in the representation space, while pushing views from different samples ("negatives") apart.

By learning to solve the pretext task, the model is forced to learn meaningful, high-level representations of the data's underlying structure. These learned representations, encoded in the model's parameters, can then be transferred and fine-tuned for a downstream supervised task where labelled data is scarce.

Further Reading The content of this section and the previous one (Section 2.3) provide a high-level introduction to the fundamental concepts and areas of machine learning that are relevant to our work. Nevertheless, there are still many further concepts and details that cannot be included within the constraints of a single thesis. For a deeper technical introduction to these topics, readers are encouraged to consult foundational texts such as Goodfellow et al. [45] and the textbook by Bishop [13]. In the following section, we will now go beyond this overview of fundamental concepts by introducing specific machine learning models that are included in the experiments of future chapters.

2.5 Overview of Machine Learning Architectures

In this section, we provide a technical background on the core machine learning models that are either foundational to this thesis or are utilised across multiple experimental chapters in order to provide a consistent reference for these key architectures. Models that are specific to a single, self-contained study are described within the methodology of their respective chapters to maintain contextual clarity. For instance, the imputation-specific deep learning models, such as BRITS and CSDI, are detailed in Chapter 6, along with simple baselines such as Linear and Cubic Spline interpolation. The architectures detailed below were chosen as they represent state-of-the-art approaches from distinct families relevant to the primary predictive tasks of this thesis.

2.5.1 Non-Deep Learning Approaches

We first explore non-deep learning approaches, where a trained model is used which does not depend on a deep neural network architecture.

ROCKET and MiniROCKET ROCKET (RandOm Convolutional KErnel Transform), proposed by Dempster et al. [28] as an "exceptionally fast and accurate" model for TSC, stands out among other state-of-the-art models in its field due to its fast training speed. Its approach,

using a large ensemble of random kernels (a "shotgun approach"), allows it to capture diverse patterns effectively across different types of sequential data without requiring extensive architecture-specific tuning or manual feature engineering. This characteristic, combined with its rapid training cycle which facilitates quick iteration and feasibility assessment on novel datasets like the laser reflection profiles used in this work, makes it a valuable tool, especially when exploring problems without established benchmarks or extensive prior feature engineering. The speed is also relevant to the industrial context in terms of enabling rapid model updates or retraining cycles.

Its innovation lies in using a large number of random, non-trainable convolutional kernels (10,000 by default), each with random parameterisation, to efficiently transform a time series into a rich feature vector. The parameters for each kernel are set as follows:

- **Length:** The kernel's length is chosen with equal probability from the set $\{7,9,11\}$
- **Weights:** The weights are sampled from a standard normal distribution, $\mathcal{N}(0,1)$, before being mean-centered (i.e. the mean of the weights is subtracted from each weight)
- **Bias:** The value for the bias term is sampled from a uniform distribution, $\mathcal{U}(-1,1)$
- **Dilation:** Dilation is sampled on an exponential scale $d = \lfloor 2^x \rfloor$ where $x \sim \mathcal{U}(0,A)$ and $A = log_2 \frac{l_{input}-1}{l_{kernel}-1}$, with l_{input} being the length of the input time series and l_{kernel} being the length of the kernel
- **Padding:** For each kernel, padding is randomly added with a probability of 0.5. If padding is used, zero-padding is added to the beginning and end of every time series when the kernel is applied

For each kernel, two summary statistics are extracted via global pooling: the maximum value (max pooling) and the proportion of positive values (PPV), resulting in 20,000 features (2 for each of the 10,000 kernels). These 20,000 features are then used to train a simple, fast linear classifier like a ridge regressor. Rocket's feature extraction process yields broad and informative features, making it adaptable for regression tasks, which can be achieved by replacing the ridge classifier with a ridge regressor. This adaptability has been demonstrated by Chang Wei Tan et al. [113], where Rocket outperformed other models and emerged as the best-performing model in their TSER experiments. In the paper, Rocket achieves high accuracy while exhibiting the shortest training phase when evaluated across the 85 UCR TSC datasets [27].

In this thesis, we make contributions to improve the efficiency of the ROCKET model. These include a GPU implementation [87], as well as a method to train an efficient subset of kernels, and are introduced in Chapter 7.

MiniROCKET, introduced by Angus Dempster et al. [29], builds upon the efficiency of the ROCKET model, further distinguishing itself from other state-of-the-art approaches in terms of speed. It offers notable advancements in accuracy relative to computational cost. This efficiency boost is achieved through modifications to the kernel parameterisation used in the original ROCKET model. By employing binary kernels with two potential values (-1 or 2), MiniROCKET performs convolution operations based on addition rather than multiplication. With only two kernel values available, there exist 84 unique kernels. To approximate the use of 10,000 kernels, these 84 kernels have 119 dilations, chosen based on the input data length, resulting in 9,996 unique kernels. Additionally, Biases are sampled from the input data, resulting in a normalisation effect. Furthermore, MiniROCKET eliminates the max pooling operation, opting for only PPV pooling, yielding one feature per kernel, amounting to approximately 10,000 features compared to ROCKET's 20,000. Like ROCKET, MiniROCKET is a transform which produces features with which to train a regression model afterwards. The authors report that MiniROCKET achieves processing speeds roughly 75 times faster on larger datasets while maintaining similar accuracy to ROCKET, a model already recognised for its efficiency compared to other models in the existing literature.

2.5.2 1D Deep Learning Architectures

Temporal Convolutional Network (TCN) The TCN, as defined by Bai et al. [8], is a convolutional architecture designed for sequence modelling. It uses dilated stacked convolutional blocks, where the dilation factor increases exponentially with each block. This structure allows the network to have an exceptionally large receptive field with a relatively small number of layers, enabling it to capture long-range temporal dependencies efficiently. Residual connections are used between layers to facilitate the training of deep networks. The Receptive Field (RF) of the model can be estimated with the following equation:

$$RF = 2 \cdot \sum_{i=0}^{n-1} (k_s - 1) \cdot d^i \tag{2.11}$$

Each convolutional block contains two convolutional layers, hence the multiplication by two in the equation. The model has two additional hyperparameters: kernel size, k_s, and the number of layers, which can also vary in the dimension of the output channels. Both of these hyperpa-

rameters can be selected to increase model complexity and to change the receptive field, as it is likely beneficial for the receptive field to encompass the data length. The model also makes use of residual connections between blocks, weight norm before convolution, and Dropout in the network.

2.6 Conclusion

In this chapter, we have introduced the relevant context which is necessary to understand the work of this thesis. This has included both understanding of the domain where we apply our research, steel manufacturing, and the specific problems we address within that domain, as well as the fundamental knowledge of machine learning which is required to define the architectures of the following chapters. We have covered only that work which we rely on for our own, and in the next chapter we expand this foundation to include a summary of research which is similar to ours while addressing either slightly different problems or incorporating different design choices. In this related work chapter, we will explore literature which encompasses the application of machine learning to steel manufacturing and surface metrology, as well as that which presents new developments in time series analysis and representation learning.

Chapter 3

Related Work

This chapter provides a review of the key literature relevant to the work presented in this thesis. The review is structured to guide the reader from the broad application domain to the specific machine learning techniques that are leveraged and advanced. We begin by establishing the industrial context, reviewing literature on surface metrology in manufacturing.

However, it is important to note that, to the best of our knowledge, no deep learning architectures have been proposed in the literature specifically for the prediction of surface roughness parameters from high-aspect-ratio optical scan data, or indeed for data of a similar nature. Consequently, there is no direct 'state-of-the-art' baseline for this novel application, and it is not immediately obvious which model families, whether image-based or time-series-based, will yield the best performance. To address this, we must look for structural similarities in other problem areas. Therefore, the latter half of this review surveys state-of-the-art architectures from the broader, adjacent fields of Computer Vision, Time Series Classification (TSC), and Time Series Extrinsic Regression (TSER). We specifically select and review models whose architectural properties make them theoretically suitable candidates for transfer to our domain, despite not being originally designed for surface metrology.

Finally, we narrow our focus to the specific deep learning sub-fields that are central to the contributions of this thesis: time series analysis (including imputation, classification, and model pruning) and representation learning with unlabelled data.

3.1 Machine Learning for Surface Metrology in Manufacturing

The increasing availability of sensor data in industrial environments has driven a widespread adoption of data-driven methods for process optimisation and quality control [72, 111]. This section reviews the literature at the intersection of machine learning and surface metrology. Crucially, it highlights a significant gap in existing research: while machine learning is used in steel manufacturing, its application to predict surface roughness from on-line optical sensor data in a continuous temper mill remains unexplored. The review first distinguishes the task of surface roughness prediction from the more common problem of visual defect detection, before surveying the various approaches that have been applied to predicting roughness parameters from different types of sensor data.

3.1.1 ML for surface prediction

Machine learning is commonly applied in manufacturing to perform automated visual inspection of surfaces [40, 65, 134]. In strip steel production, for example, these techniques are used for the detection of macroscopic defects. Such systems typically use high-resolution cameras to capture images of the surface during production and apply machine vision or deep learning models to locate or classify anomalies like scratches, holes, or inclusions, which are small, non-metallic particles trapped within the steel. This body of work is important for industrial quality control and shares challenges with our setting, such as the requirement for high-speed processing. However, its limitation, in the context of our research, is that it is fundamentally distinct: it focuses on classifying discrete, large-scale defects, not on characterising the statistical properties of the entire surface's micro-scale texture. The novelty of this thesis, therefore, is to regress a continuous statistical parameter (R_a) from this micro-texture, a task these visual inspection methods are not designed for.

3.1.2 Machine Learning for Steel Surfaces Roughness Prediction

While machine learning is now widely applied across manufacturing [110], its use for predicting surface roughness parameters is a fragmented field of research, with studies spread across various domains. The existing literature largely focuses on manufacturing processes distinct from the high-speed rolling of steel strip, such as machining, milling, and additive manufacturing. This section reviews these applications to establish the broader context and, in doing so, highlights the specific research gap that this thesis addresses.

A large proportion of the related work focuses on predicting surface roughness from auxiliary sensor data or pre-defined process parameters, rather than from a direct measurement of the surface itself (such as optical measurements). For instance, in the context of machining, Elangovan et al. [37] use multiple regression on statistical features (e.g., mean, skewness) extracted from tool vibration signals, alongside machining parameters such as tool wear and speed, to predict the R_a of metal objects turned on a lathe. Although this approach is practical, more informative features could potentially be extracted using more advanced signal processing or deep learning techniques. Similarly, other research has used neural networks to model the relationship between Computer Numerical Control (CNC) steel milling parameters (such as material removal rate) and the resulting R_a [120]. The Ra parameter has also been used in multiple regression analysis to understand the influence of different parameters in the dressing (replenishing) grinding wheels [88]. The key limitation of these approaches is that they model correlations with process parameters or indirect signals (like vibration), not direct surface measurements, making them susceptible to unobserved process variations. The novel contribution of this thesis, therefore, lies in using high-resolution, direct optical sensor data from the steel coil, which contains richer, more direct information about the surface micro-texture, as the primary input for R_a prediction.

The prediction of surface roughness parameters is also a significant area of research in additive manufacturing (AM), where achieving the desired surface quality without costly and time-consuming post-processing is a major challenge. Numerous studies have applied AI and machine learning to predict final surface roughness from process parameters, aiming to reduce the need for experimental trial-and-error. For example, Koo et al. [68] developed machine learning models, including Support Vector Regression (SVR) and Random Forest (RF), to predict the downskin surface roughness of components fabricated via laser powder bed fusion (PBF-LB). Their models used process parameters such as laser power and scanning speed, alongside the part's overhang angle, as inputs to predict the final R_a value. Across the AM literature, many other studies apply similar principles, predicting the final surface quality in processes like 3D printing, where R_a can be correlated with material strength. For example, Mushtaq et al. have published several works discussing the importance of R_a in 3D printing, where it can be correlated with other desirable properties like material strength [89, 90]. For instance, with a too fast print speed, the material has less time to settle and solidify before adding the next layer resulting in rougher surface quality and decreased strength. Also, La-Pérdomo et al. [71] use a Multi-layer Perceptron (MLP) and an adaptive neuro-fuzzy inference

system for roughness prediction in selective laser melting.

While valuable, the limitation of these AM-focused studies is their inapplicability to our problem. Their models are tailored for the static, layer-by-layer nature of additive manufacturing, which is fundamentally different from the continuous, high-speed dynamics of continuous strip temper rolling. Our work provides a novel application by developing models that can operate on-line, processing a continuous stream of optical sensor data from the steel coil, a challenge not addressed by the AM literature.

The availability of fast, on-line monitoring opens the door for real-time closed-loop control systems, which can automate temper mill adjustments to meet specific customer requirements. The work most closely related to this industrial motivation is by Cheri et al. [26], who proposed an on-line intelligent control method for cold-rolled strip steel. Their system uses on-line measurements to inform a fuzzy neural network that, in turn, adjusts the roll force in a closed-loop control system. However, a critical distinction is that their system does not predict the R_a value but instead relies on the intrinsic accuracy of the on-line measurement system as its ground truth. The work in this thesis addresses a more fundamental problem: the inherent inaccuracy of such on-line optical systems when compared to the stylus-based metrology standards used for final quality assurance. We use data from an on-line measurement system and develop machine learning models that transform the raw optical readings into a more accurate prediction that aligns with the gold-standard stylus measurement. This step is a prerequisite for any high-fidelity control system of the type they propose.

Other related research in steel manufacturing demonstrates the broader trend of applying machine learning to the domain, even if not focused on roughness. Gupta et al. [48] used a k-nearest neighbour (k-NN) algorithm for the multiclass classification of steel microstructure types based on composition and heat treatment data. More recently, Sawai et al. [106] used a convolutional neural network (CNN) for image regression to directly link microstructural images with mechanical properties, employing an explainable AI approach with UMAP dimensionality reduction to highlight important features of the steel. These studies confirm the viability of data-driven approaches in steel science, but the specific, novel contribution of this thesis remains the application of deep learning to the on-line prediction of surface roughness from optical sensor data.

Regression has also been applied to other practical use cases including those which require the use of other types of data beyond images or time series. These applications span diverse domains: energy sector optimisation where gradient boosting regression models optimise

tight gas reservoir fracturing parameters using geological and operational data [125], emissions monitoring where regression models predict engine pollutant outputs from combustion parameters [33, 92], materials engineering where nonlinear regression predicts polymer mechanical properties from compositional variables [83], and healthcare analytics where regression-based models support clinical decision-making using patient data [97] . This demonstrates the versatility of regression approaches across fields requiring predictive modelling with structured, non-temporal data inputs.

3.2 Advances in Deep Learning for Time Series Analysis

The sensor data at the heart of this thesis, while possessing a two-dimensional spatial structure, can also be naturally treated as a multivariate time series due to its extreme aspect ratio (e.g., $20 \times 50,000$). To the best of our knowledge, no established benchmark architecture exists for this specific data modality. As a result, there is no direct state-of-the-art baseline for this application, necessitating the evaluation of architectures drawn from structurally related research fields.

To address this, we survey architectures from the broader, adjacent fields of Computer Vision and Time Series Classification (TSC). We specifically review models—such as Temporal Convolutional Networks (TCNs) and Inception-based ensembles, whose architectural properties (e.g., effective receptive field size, multi-scale feature extraction) make them theoretically suitable candidates for transfer to our domain.

Broadly, the state-of-the-art in these fields has shifted towards two primary families:

- **Convolutional Architectures:** 1D-CNNs, such as Temporal Convolutional Networks (TCNs) [8], ResNet [52], and InceptionTime [59], have established themselves as the performance standard for classification. Their use of deep, dilated, or multi-scale convolutions allows them to capture complex local surface textures efficiently.

- **Transformers and Attention:** For tasks requiring global context, such as filling large gaps, Transformer architectures have become dominant [32]. By utilising self-attention, models like SAITS [35] can effectively capture long-term dependencies across the entire sequence length, often surpassing recurrent models on long-horizon time series tasks [32].

This section reviews the literature in three key areas that directly inform the contributions of this thesis. First, we discuss the challenges and competing paradigms of modelling high-aspect-ratio data. Second, we review state-of-the-art methods for time series imputation. Finally, we survey the evolution of time series classification, with a specific focus on the ROCKET family of models and their subsequent pruning.

3.2.1 Learning with High Aspect Ratio Data

Many real-world datasets, particularly from scientific and industrial sensors, do not conform to the near-square formats typical of common computer vision benchmarks. These datasets often exhibit a high aspect ratio, where one dimension is substantially larger than the other. Examples from other domains include panoramic images, dental radiographs [43], and whole-slide histopathology images, which present challenges for memory and computation [122]. The laser reflection data central to this thesis represents an extreme case of this, forming long, narrow two-dimensional arrays that encode structured spatial information but lack the object-based semantics of conventional images.

This unique structure can pose a challenge for standard deep learning models and gives rise to two competing modelling paradigms. One approach is to adapt vision models. Architectures like 2D Convolutional Neural Networks (CNNs) [74] or Vision Transformers (ViTs) [34], which process data as a sequence of smaller patches and can split a long input array into a sequence of fixed-size patches, can be applied. However, these models are often more computationally complex and are typically designed for square-like inputs. Therefore, if not designed carefully, they can struggle to capture long-range dependencies across the elongated axis. An alternative and often more natural approach is to treat the data as a multivariate time series (where the narrow axis represents the channels or variables) and apply models from the Time Series Classification (TSC) and Regression (TSER) literature [59, 113]. Models such as Temporal Convolutional Networks (TCNs) [8, 73] are particularly well-suited, as they are specifically designed to efficiently model very long sequences.

In the domain of Brain-Computer Interfaces, several studies [62, 67, 108] delve into machine learning for electroencephalogram (EEG) datasets, primarily aimed at classifying actions based on brain signals. Although these are classification problems, they share a resemblance with our regression problem, as they involve multichannel signals collected from multiple sensors spanning a substantial number of time steps, making them another example of high-aspect-ratio data. A key distinction, however, lies in the spatial topology of the channels. The rela-

tionship between EEG sensors is defined by their irregular physical placement on the scalp, which contrasts with the fixed, grid-like structure of the optical sensor data used in this work. Other literature has explored the importance of maintaining aspect ratio for learning in other domains, such as the classification of cancer cell images [76].

3.2.2 Time Series Imputation

Missing data is a pervasive problem in real-world time series, arising from sensor failures, transmission errors, or data corruption. Classical statistical methods provide a foundational toolkit for addressing this issue. Simple approaches include mean, median, or Last Observation Carried Forward (LOCF) imputation, while more sophisticated techniques involve fitting a model to the observed data. These include interpolation methods, such as linear or spline interpolation, and model-based approaches that leverage the temporal structure, such as AutoRegressive Integrated Moving Average (ARIMA) models, to forecast the missing values, as reviewed in [49].

While classical methods are effective for certain data types, they often struggle to capture the complex, non-linear dynamics present in modern, high-dimensional datasets. This has motivated a shift towards deep learning approaches for imputation. Recurrent Neural Networks (RNNs), particularly architectures like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs), were early successes in this domain. These models process data sequentially, using an internal state to model temporal dependencies. A notable advancement in this area is the Bidirectional Recurrent Imputation for Time Series (BRITS) model [16], which processes the time series in both forward and backward directions and incorporates a temporal decay mechanism to weight the influence of observations based on their proximity to the missing values.

The introduction of the attention mechanism, particularly within the Transformer architecture, marked a significant leap forward for sequence modelling. Unlike RNNs, which can struggle with very long-range dependencies, attention allows a model to directly weigh the importance of all other time steps when making a prediction for a specific point. This paradigm was effectively adapted for imputation in the model: Self-Attention-based Imputation for Time Series (SAITS) [35]. SAITS employs a masked self-attention mechanism and a joint optimisation objective that combines a masked reconstruction loss with a contrastive loss, enabling it to capture both temporal and feature-wise relationships with high fidelity.

More recently, generative models have emerged as a powerful alternative for imputation,

treating the task as a conditional data generation problem. Diffusion models, which learn to reverse a process of gradually adding noise to data, have been adapted for this purpose. Conditional Score-based Diffusion Models for Imputation (CSDI) [115], for example, can generate highly realistic imputations by conditioning the denoising process on the observed data points. Generative Adversarial Networks (GANs) have also been applied, where a generator network learns to produce plausible values for missing segments, and a discriminator network tries to distinguish between real and imputed data [128].

To the best of our knowledge, very little work in the imputation literature explicitly addresses the preservation of statistical texture for surfaces. The most directly relevant work is by Jawaid et al. [60], who propose using Gaussian Processes for imputing spurious data on technical surfaces with the explicit goal of "maintaining the surface characteristics". Their approach, which involves sampling from a posterior distribution, is conceptually designed to generate realistic textures. However, their analysis is primarily qualitative, and the authors note that a quantitative, comparative study against other data imputation techniques is a subject for future work.

Despite the increasing sophistication of these models, their development and benchmarking have been overwhelmingly driven by a single objective: minimising pointwise reconstruction errors like Mean Squared Error (MSE) or Mean Absolute Error (MAE). This paradigm is evident in recent, comprehensive benchmarking studies focused on standard data archives. For example, an extensive review by Kazijevs et al. [63] benchmarked numerous state-of-the-art deep imputation methods across five different time series health datasets. For their primary evaluation of imputation accuracy, they relied exclusively on Normalised Root Mean Squared Deviation (NRMSD), a variant of MSE and therefore a pointwise loss. While the authors also assessed the utility of the imputed data via a downstream classification task, the core evaluation of the imputation itself remained rooted in pointwise error, with no analysis of whether the methods preserved the underlying statistical or distributional properties of the signals.

Some specialised fields have adopted more statistical metrics. For instance, in signal processing, Sánchez et al. [104], faced with reconstructing a time signal from only its power spectral density (PSD), use a spectral loss function for training and evaluate success based on a downstream system identification task, completely avoiding time-domain RMSE. This highlights that alternatives to pointwise error exist, yet they are less common.

This narrow focus on pointwise accuracy persists even in advanced, domain-specific research. In industrial settings, for example, Jiang et al. [61] proposed a novel framework

(PFIDM) for industrial soft sensor modelling that uses a diffusion model to impute missing data, with a feedback mechanism from the downstream task guiding the imputation. Notably, while the authors provide a visual comparison of the data's density distributions, implying an awareness of the importance of preserving statistical properties, their formal, quantitative evaluation of both the imputation and the final soft sensor was still based exclusively on Root Mean Square Error (RMSE) and Mean Absolute Error (MAE).

This trend of relying on pointwise metrics holds even in studies that explicitly aim to preserve statistical properties. For instance, Ahmad et al. [4] introduce a novel method that uses periodically correlated components (VBPBB) as auxiliary variables in an imputation model, with the stated goal of "preserving key statistical properties" . To support this claim, the authors provide a visual comparison of the imputed time series against the original signal and perform a correlation analysis. However, the primary quantitative results presented to demonstrate their method's superiority are based exclusively on a reduction in Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) . The same is true for a robust univariate imputation algorithm (UIM) from Han et al. [49] designed for wastewater treatment process data. Their novel method decomposes the time series into trend, seasonal, and remainder components and imputes each separately. Yet, their evaluation of this complex approach was limited to two pointwise metrics: RMSE and a custom "Similarity" score, which is a normalised metric based on absolute error, fundamentally a pointwise error. Similarly, Zhang et al. [130] propose a sophisticated, physics-informed model for radiomap inpainting with the goal of reconstructing a realistic signal field, preserving the "texture" of the radiomap is implicitly important. Their methods are designed to propagate patterns and structures into the missing regions. However, again, despite the sophisticated, physics-informed methodology, the quantitative evaluation relies entirely on standard pointwise error metrics.

While our review confirms that the vast majority of the imputation literature relies on pointwise error metrics, a few notable exceptions in specialised domains highlight the need for more appropriate evaluation. These exceptions, however, tend to fall into two categories: those that operate in a transformed domain, and those whose model architecture is inherently probabilistic. For instance, in signal processing, Sánchez et al. [104] developed a method that operates in the frequency domain, using a spectral loss function to impute phase information from a signal's power spectral density. Other approaches achieve texture preservation through their intrinsic model structure. Generative models, such as the diffusion-based CSDI [115] and the PFIDM model by Jiang et al. [61] (which includes a KL divergence term in its loss), are de-

signed to learn and replicate the underlying data distribution. Similarly, the stochastic process approach of Jawaid et al. [60], which uses Gaussian Processes, imputes missing surface data by sampling from a learned posterior distribution. A significant gap therefore remains in making and adapting powerful, general-purpose, time-domain architectures, such as Transformer-based models like SAITS, which are typically trained with pointwise loss, to preserve statistical texture directly. To the best of our knowledge, this has not been explored. This is a critical area for investigation, as transforming data to other domains can be lossy, and generative models may have different architectural constraints compared to state-of-the-art predictive models.

This evaluation gap is a critical limitation, as the primary goal in many industrial applications is not perfect pointwise reconstruction, but the preservation of the signal's statistical integrity. This requirement is often overlooked even in closely related domains; for instance, a comprehensive review of AI-driven soft sensors for process industries by Perera et al. confirms that performance is almost universally measured by prediction accuracy metrics like RMSE and MSE, rather than by the preservation of statistical properties [96]. Therefore, while a handful of domain-specific methods that assign value to signal characteristics exist, the development of a general deep learning framework for time series imputation that is explicitly and quantitatively evaluated on its ability to preserve statistical texture remains, to the best of our knowledge, a novel contribution. This is especially true within the specific context of industrial surface metrology, where, to our knowledge, no prior work has any form of imputation with the aim to preserve surface texture parameters such as R_a. This thesis aims to address that gap by proposing and validating such a framework, using the challenging domain of industrial surface metrology as its primary application and testbed.

3.2.3 Time Series Classification and the ROCKET Family

As mentioned before, the sensor data in this thesis can be modelled as a multivariate time series. While for our specific data the task is regression, the architectural advancements in time series are largely driven by the more mature and intensely studied field of Time Series Classification (TSC) and not regression. A significant catalyst for progress in TSC has been the UCR Time Series Archive [27], a comprehensive collection of datasets that has become the de-facto standard for benchmarking new algorithms. However, this archive has notable limitations; it predominantly contains datasets that are short, univariate, and have a small number of samples.

This characteristic can favour classical algorithms over deep learning models, which typically require more extensive data to learn effectively. This profile is a key limitation of the

UCR-driven literature, as its focus on short, univariate classification problems contrasts sharply with our steel coil data, which is a long, multivariate regression problem. A contribution of this work is therefore the adaptation and evaluation of these state-of-the-art classification-focused architectures for this specific industrial regression (TSER) task.

Historically, TSC has been dominated by distance-based methods, with Dynamic Time Warping (DTW) being a prominent baseline. The field has since evolved to include a wide array of approaches, including dictionary-based methods, shapelet-based models, and, increasingly, deep learning architectures. Deep learning models such as InceptionTime [59], which adapts the successful Inception architecture from computer vision for time series, achieved state-of-the-art accuracy when they were published by automatically learning hierarchical features and utilising ensembles. It should be noted that the field has evolved into powerful ensemble models [7] containing many of the smaller works within, and therefore, they are extremely computationally expensive to train.

The high computational cost of training the large ensembles that dominated the TSC benchmarks motivated a search for more efficient alternatives, leading to the ROCKET (RandOm Convolutional KErnel Transform) [28]. When published, this model achieved statistically insignificant differences in accuracy from the top model while being exceptionally efficient to train. ROCKET is a feature transformer, in contrast to deep learning models that learn their convolutional filters, ROCKET applies 10,000 fixed, non-trainable convolutional kernels with random parameterisation. For each kernel, two features are extracted via global pooling: the maximum value (max pooling) and the proportion of positive values (PPV). The resulting 20,000 features are then used to train a simple, fast linear classifier, such as a ridge classifier. While the model has great training efficiency, it is worth noting that during inference, applying the large bank of 10,000 kernels will be slower in most cases than using a single, compact deep learning model as these typically have less kernel filters. Its effectiveness has been confirmed on multivariate datasets where it was found to have the best average rank among top-performing algorithms [101]. In a testament to its impact on the field, ROCKET and its successors were subsequently incorporated into the next generation of state-of-the-art ensemble models, continuing the cycle of innovation on the UCR benchmarks [84].

The success of ROCKET has spawned a family of related models. MiniROCKET [29] was introduced to further improve computational efficiency. By using a fixed set of 84 binary kernels convolved via addition instead of multiplication and using only the PPV feature, MiniROCKET was shown to be up to 75 times faster than the original ROCKET with compa-

rable accuracy. Subsequently, MultiRocket [114] demonstrated how the efficiency gains from MiniROCKET could be leveraged to increase model complexity and push for even higher accuracy, for instance by applying transforms to the input data and using additional pooling operations. The principles of these models have also been shown to be highly effective when adapted for Time Series Extrinsic Regression (TSER) tasks, as demonstrated by Tan et al. [113].

While the ROCKET family demonstrates the power of random features, the use of such a large, fixed set of kernels raises questions about efficiency and redundancy, motivating investigations into model pruning. Several distinct methodologies have been proposed to identify and remove non-essential kernels. The pioneering work in this area, S-ROCKET, frames kernel selection as a combinatorial optimisation problem [103]. It employs a population-based evolutionary algorithm to search for an optimal subset of kernels that preserves classification accuracy while maximising sparsity.

A different approach, POCKET, reframes the problem from a feature selection perspective [17]. It treats the features generated by each kernel as a group and applies group elastic net regularisation to the linear classifier, which encourages the weights for entire feature groups to become zero, thereby identifying their corresponding kernels for pruning.

A third strategy is employed by Detach-ROCKET, which uses a sequential backward-elimination method called Sequential Feature Detachment (SFD) [119]. This technique iteratively retrains the classifier, ranks the importance of each feature by the magnitude of its learned coefficient, and prunes the least influential ones.

All three approaches successfully demonstrate that a large fraction of ROCKET's kernels, often over 60%, can be removed with little to no loss in accuracy, significantly improving inference efficiency [17, 103, 119]. These works focus on creating more efficient models by pruning the fixed, random kernels, operating entirely within the original ROCKET paradigm where the kernels themselves are never trained. The novel contribution of this thesis, therefore, is to challenge this core assumption. Rather than proposing a new pruning technique, this work investigates the possibility of the subsequent step of fine-tuning these traditionally non-trainable kernels after pruning, in an attempt to recover or even exceed the performance of the original, larger model. This initial pruning stage is a critical prerequisite, as attempting to fine-tune the entire initial set of 10,000 random kernels would be computationally infeasible. Also, the ROCKET model is not differentiable, which is required for training with backpropagation.

3.3 Representation Learning with Unlabelled Data

In industrial contexts such as steel manufacturing, obtaining large amounts of high-quality labelled data can be prohibitively expensive or as is our case, impractical. Unfortunately however, modern machine learning techniques require vast amounts of data in order to perform robustly and produce accurate predictions. Representation learning approaches, including self-supervised learning, are one avenue to address this challenge by allowing models to learn useful feature representations from raw, unlabelled data, which is typically more abundant in industrial settings. At a high level, these approaches employ pretext tasks which generate supervisory signals from the data itself to enable the extraction of generalisable patterns, which enables models to perform more adaptably and accurately when labelled data is scarce.

A thorough review of self-supervised learning (SSL) methods is provided in the survey by Gui et al. [47]. One of the key areas explored is the concept of Contrastive Learning (CL) which compares unlabelled instances to eachother. Initially, this approach built on the task of instance discrimination [123, 132], where the aim is to generate distinct views of a single data point. The model should learn to pull together positive pairs of the same data instance in the embedding space, while pushing apart representations of different instances (negative pairs).

More recent key approaches within this area include MoCo v1 [50], MoCo v2 [23], SimCLR v1 [19] and SimCLR v2 [21]. With the introduction of MoCo, contrastive learning moved away from instance discrimination, instead making use of a dictionary look-up task. The contrastive loss function is low when a query q is similar to its corresponding key k and dissimilar to all other keys. Meanwhile, SimCLR introduces a mini-batch sampling strategy with N data instances. A contrastive training task is then formulated on pairs of these instances, with the remaining ones treated as negative examples. Notably, both MoCo and SimCLR rely on data augmentation to generate positive examples, including cropping, resizing and the use of colour. These models have shown extremely promising performance for self-supervised pre-training, almost matching the performance of supervised learning. Approaches have also been developed specifically for time series data, including TimeCLR [127], TS2Vec [129], TF-C [131], BTSF [126], and MF-CLR [36].

Aside from these contrastive learning approaches, generative models such as BEiT [10], Contextualised Learning [22], and Masked Image Modelling [124] are another family of approaches. These typically make use of an autoencoder architecture which utilises co-occurrence relationships among image patches to perform a supervised training task. These approaches were initially applied to language tasks [15, 31], but were adapted for vision tasks

with the introduction of the Vision Transformer (ViT) [34].

The primary limitation of this existing work is one of application domain. While these methods have proven highly effective, their development and benchmarking have been overwhelmingly focused on standard academic datasets (e.g., ImageNet, UCR time series archives) which consist of natural images, audio, or general sensor data. To the best of our knowledge, these state-of-the-art self-supervised learning techniques have not been applied to or evaluated on such high-aspect-ratio data and especially not on the unique, non-semantic optical sensor data used in this thesis. The novel contribution of this thesis is therefore to address this gap by providing the first investigation into these powerful pre-training strategies on high-aspect-ratio, non-semantic data, using our specific industrial context to apply them to the unlabelled steel coil data and evaluate their impact on the sample efficiency and final accuracy of the downstream R_a regression task.

3.4 Conclusion

In this chapter, we have expanded on the context provided in the previous chapter by additionally considering the related work which addresses similar problems to ours, or introduces relevant models or concepts upon which we build. This has included other literature which applies machine learning to surface metrology in manufacturing, especially to surface roughness estimation, and particular work advancing deep learning for data similar to ours. In particular, time series analysis is a field upon which we can draw due to the long and thin spatial nature of our steel surface data. Both imputation and representation learning are specific areas of machine learning explored due to their relevance for our experiments in later chapters.

Having reviewed the existing literature and identified broad research areas, a detailed understanding of the specific industrial data is now necessary before the precise research problems addressed in this thesis can be fully formulated. Therefore, the next chapter transitions to this crucial step, providing a summary of the data we make use of for our experiments. This summary will address the primary data sources gathered using industrial hardware, detail the baseline calculation method, and outline the inherent data challenges which motivate, along with the gaps in the related work, the subsequent problem statement (Chapter 5).

Chapter 4

Data Acquisition and Datasets

This chapter details the origin, characteristics, and curation of the datasets that form the empirical foundation of this thesis. The research leverages two primary sources of raw data: high-dimensional laser reflection profiles from an industrial sensor, and high-fidelity surface height profiles from conventional stylus profilometry. While the data acquisition hardware and initial sample measurements pre-dated this work, we must transform this raw information into structured, curated datasets designed specifically to address the machine learning challenges central to this research.

The data was originally collected by our industrial partners to investigate discrepancies between the sensor's physics-based output and the stylus ground truth under controlled lab conditions. This initial analysis confirmed that the laser measurement machine's standard calculation was insufficient for production use, thereby motivating the data-driven methods and approaches explored in this thesis. This research is therefore built upon two primary collections of raw data, hereafter referred to as the primary **Data Sources**: 1) a *Lab Data Source*, containing partially linked laser and stylus measurements from physical steel samples, and 2) an *In-Production Data Source*, comprising a larger volume of laser-scan data captured during live manufacturing.

From these two primary sources, several distinct **ML Datasets** are curated throughout this thesis to address specific research questions. This chapter focuses on describing the origin, structure, and key characteristics of the primary Data Sources themselves. It details the acquisition hardware, a backwards estimation of how the baseline physics-based closed-form solution is calculated, and the fundamental challenges inherent in the data. The specific construction of ML datasets for tasks such as direct roughness prediction, time-series imputation, or self-

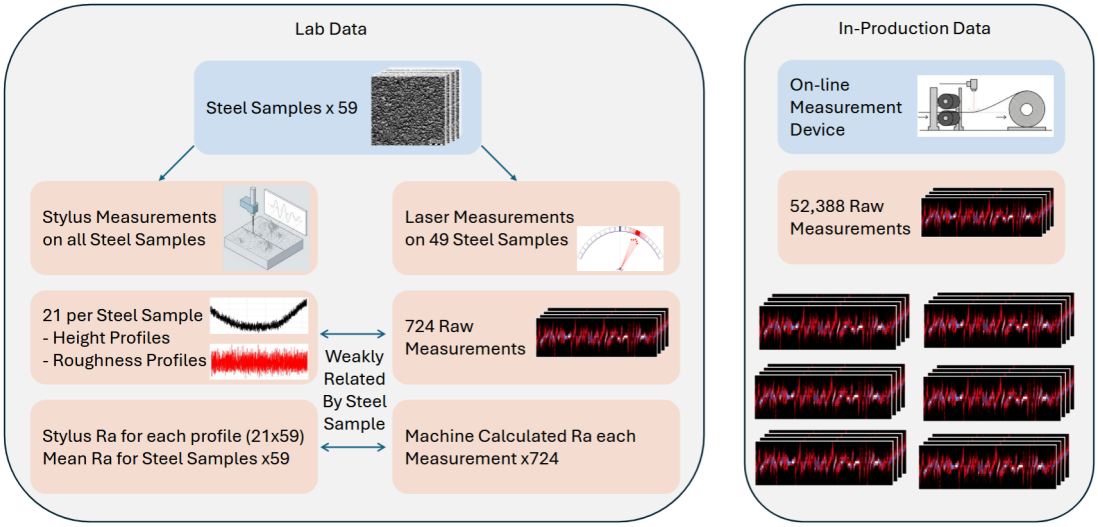


Figure 4.1: An overview of the data used in this thesis, showing the relationship between the Lab Data and the In-Production Data. The Lab Data section illustrates the weakly related correspondence between stylus-based “ground truth" measurements and the laser-based measurements for each steel sample. The In-Production section highlights the much larger volume of unlabelled laser data collected directly from the manufacturing line.

supervised pre-training is detailed within the methodology of the relevant content chapters. Due to commercial sensitivity, the underlying data is proprietary and not publicly available. However, a detailed description of the data collection process, structure, and key characteristics is provided in this chapter to support understanding and reproducibility of the research methodology.

4.1 Data Sources and Acquisition Hardware

The two primary Data Sources central to this thesis are generated by distinct measurement instruments, each providing a different perspective on the steel’s surface topography. A comprehensive technical description of these instruments is provided in Section 2.1.3 of Chapter 2; this section serves as a brief, contextual summary.

The first source of data comes from the **SORM 3plus**, an industrial non-contact sensor that characterises the surface by directing a laser at it and capturing the angular distribution of the reflected light. This is performed by a semicircular array of 20 discrete photosensors. Although the resulting data are two-dimensional (sensor intensities $\times$ time), the physical measurement itself follows a single one-dimensional track along the steel surface, as the laser illuminates

only one line and no information is captured across the transverse direction of the strip. A single measurement, or timestep, produces a 20-dimensional vector of light intensities. By taking measurements at fixed spatial intervals as the steel moves, the device generates the raw data for this thesis: a high-dimensional laser reflection profile represented as a $20 \times T$ integer matrix, where T is the number of timesteps.

The second data source, which provides the ground truth for surface roughness, is a conventional contact-based stylus profilometer. This instrument physically traces a microscopic diamond-tipped stylus across the steel's surface to measure its height variations with high fidelity. The output is a one-dimensional vector representing the surface height profile, from which standard roughness parameters, such as the arithmetic mean roughness (R_a), are calculated according to ISO standards.

4.2 The Physics-Based Closed-Form Solution

Before describing the curated data sources, it is crucial to establish how the laser measurement device calculates the surface parameters required in steel manufacturing. The SORM 3plus device uses its own proprietary, physics-based closed-form solution to convert the raw $20 \times T$ laser-reflection matrix into a single R_a value. As this algorithm is not public, the process was engineered to the best of our ability based on supplementary documentation to develop understanding of the origins of the values we use as baselines for this thesis. This section details the multi-step process of this solution for demonstration purposes, however we do not use our calculated values for any of the experiments in the thesis. The performance gap between this baseline and the ground-truth stylus measurements is the primary motivation for employing the data-driven machine learning models investigated in the subsequent chapters.

Thresholding. The first step in the process is to pre-process the raw laser intensity matrix, denoted as $X \in \mathbb{N}^{20\times T}$, where each element $X_{i,j}$ represents the 8-bit integer intensity value from sensor i at timestep j. A thresholding operation is applied to each of the 20 sensor channels independently to remove low-level noise. This can be expressed as:

$$\tilde{X}_{i,j} = (X_{i,j} - \min_{\theta}(X_{i,:}))_{+} \tag{4.1}$$

where $\min_{\theta}(X_{i,:})$ represents the noise threshold for sensor i. This value is determined by first creating a sorted list of the unique intensity values for that sensor, and then selecting the value at the θ-th index of this list. The notation $(\cdot)_{+}$ is the positive-part function. This operation

uses the threshold as a new zero-point: the threshold value is subtracted from all intensity readings in the channel, shifting the distribution downwards, and any resulting negative values are clipped to zero. An example of the effect of this thresholding is shown in Figure 4.2.

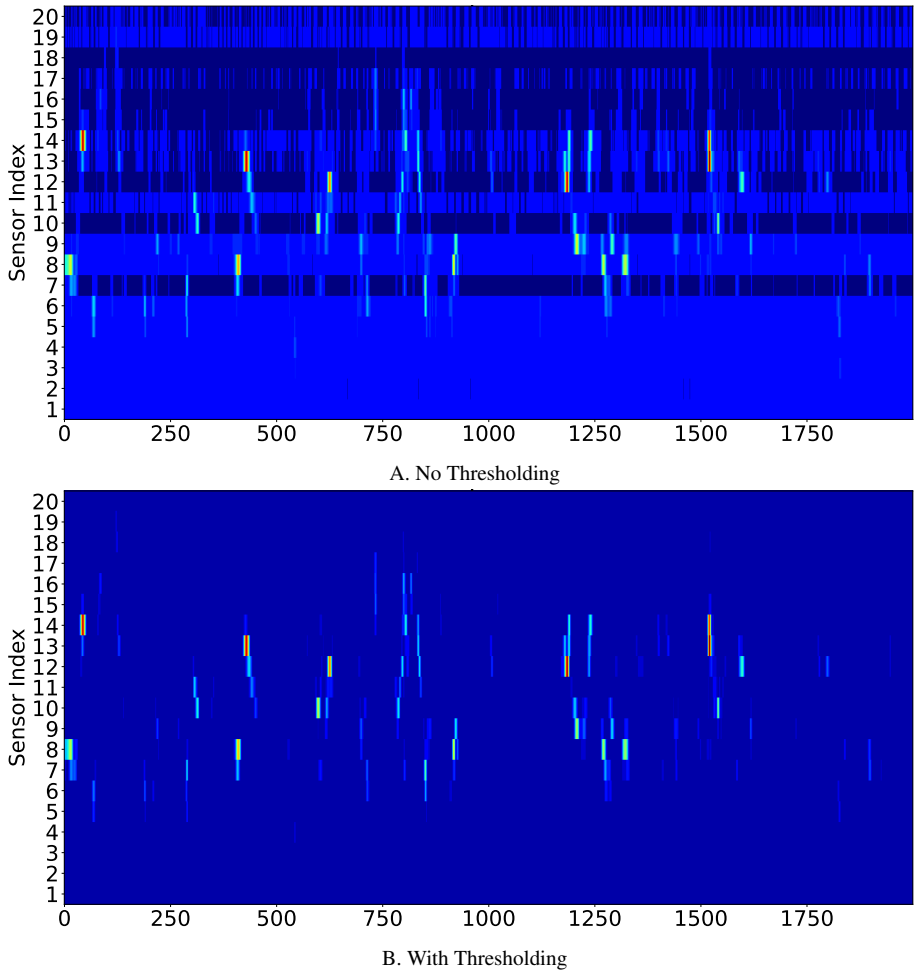


Figure 4.2: Comparison of raw intensity data for the worst outlier with and without thresholding. This is a laser measurement image from one of the MZ samples, chosen to show a difference from the normal samples.

Surface Angle Calculation. Next, a surface angle, $s_j \in \mathbb{R}$, is calculated for each timestep j. While the input intensities $\tilde{X}_{i,j}$ are integers, the calculation involves division and results in floating-point values. This is achieved by computing a weighted average of the fixed sensor angles, effectively finding the central angle of the reflected light's intensity distribution. The weights are the thresholded intensity values at that timestep. Given a constant vector A con-

taining the 20 fixed sensor angles (symmetrically spaced from -63.7,-57.0... to ...57.0,+63.7 degrees), the surface angle is calculated as:

$$s_j = \frac{1}{2} \cdot \frac{\sum_{i=1}^{20} \tilde{X}_{i,j} \cdot A_i}{\sum_{i=1}^{20} \tilde{X}_{i,j}} \tag{4.2}$$

The division by the sum of intensities makes the term a weighted average, which represents the mean angle of the reflected light relative to the incident beam. The multiplication by $\frac{1}{2}$ is a correction factor derived from the law of reflection. Assuming a near-normal incident laser beam, the angle of the surface normal relative to the beam is equal to the angle of incidence (θ_i). The law of reflection states that the angle of incidence equals the angle of reflection ($\theta_r = \theta_i$). The total measured angle of the reflected light is the sum of both ($\theta_i + \theta_r = 2\theta_i$). Therefore, to find the true angle of the surface normal, one must halve the measured angle of the reflected light. For timesteps where all sensor readings are zero, this calculation results in missing values, which are filled using linear interpolation.

Profile Integration (Gradient Calculation). The time series of surface gradients, given by the tangent of the surface angle, is then integrated to reconstruct the physical surface height profile. This is done by accumulating the gradient values, scaled by the spatial interval, Δt (0.8 μm for the lab data), as shown in:

$$\text{surface}_i = \sum_{j=0}^{i} \tan(s_j)\Delta t \tag{4.3}$$

where surface_i is the reconstructed surface height at the i-th timestep.

Filtering and R_a Calculation. The integrated surface profile is a height profile of the steel. However, we are interested in parameters calculated from the roughness profile. The height profile contains both long-wavelength (waviness) and short-wavelength (roughness) components. To isolate the roughness, a filtering procedure, detailed in Algorithm 4.1, was implemented (note that a cut-off wavelength value of 0.8mm is used for the expected R_a between ($0.1 < R_a \leq 2.0$) as in our case). As no suitable public implementation was identified, a custom implementation was developed, which uses a frequency-domain filter instead of a Gaussian filter specified in the formal ISO standard [1]. The final roughness profile is obtained by subtracting the resulting waviness profile from the original surface profile. The arithmetic mean

[1] The Python code for this FFT-based filtering method is available at: `https://github.com/alexander-milne/fft-waviness-filter`

roughness, R_a, is then calculated from this roughness profile, as defined in the Background Chapter (Section 2.2.2).

Algorithm 4.1 FFT-Based Waviness Profile Calculation

Input: Surface profile P, sample distance Δt, cutoff wavelength λ_c
Output: Waviness profile W
$P_{flipped} \leftarrow \text{reverse}(P)$
$P_{padded} \leftarrow \text{concatenate}(P_{flipped}, P, P_{flipped})$ ▷ Pad profile to mitigate edge effects
$N \leftarrow \text{length}(P_{padded})$
$k_{cutoff} \leftarrow \text{round}((N \cdot \Delta t)/\lambda_c)$ ▷ Compute FFT index corresponding to λ_c
$k_{cutoff} \leftarrow \max(1, k_{cutoff})$
$F \leftarrow \text{FFT}(P_{padded})$ ▷ Transform to frequency domain
$F[k_{cutoff} : -k_{cutoff}] \leftarrow 0$ ▷ Symmetric low-pass truncation
$W_{padded} \leftarrow \text{real}(\text{IFFT}(F))$ ▷ Transform back to spatial domain
$W \leftarrow W_{padded}[\text{length}(P_{flipped}) : -\text{length}(P_{flipped})]$ ▷ Remove padding
return W

The slice notation $[k_{\text{cutoff}} : -k_{\text{cutoff}}]$ follows Python indexing conventions, where a negative index refers to a position counted from the end of the array rather than indicating a negative frequency. In the discrete Fourier transform of a real-valued signal, the low-frequency components appear at the beginning of the array (including the DC component at index 0) and are mirrored at the end due to conjugate symmetry. The operation `filtered[k:-k] = 0` therefore sets the central high-frequency components to zero while retaining the low-frequency components at both ends of the spectrum. This implements a symmetric brick-wall low-pass filter, producing the waviness profile. The roughness profile is subsequently obtained by subtracting this waviness component from the original signal. In implementation, this operation is performed using NumPy's `fft` and array slicing functionality (e.g. `np.fft.fft`).

Baseline Performance and Motivation. As a validation of this process, Figure 4.3 compares the R_a values produced by our implementation against the values output by the SORM 3plus device itself for every scan in the Lab Data Source. The results show a very strong correlation, confirming our implementation is a faithful reproduction of the device's internal algorithm. However, a small number of significant outliers are present, which are likely due to minor differences in proprietary pre-processing steps. A group of the outliers are the lower-quality readings from the Magizinc (MZ) samples (as seen in the thresholding example Figure 4.2). Due to these minor discrepancies, the laser-based device's own reported R_a value is used as the official baseline for all subsequent experiments in this thesis. The purpose of this section has

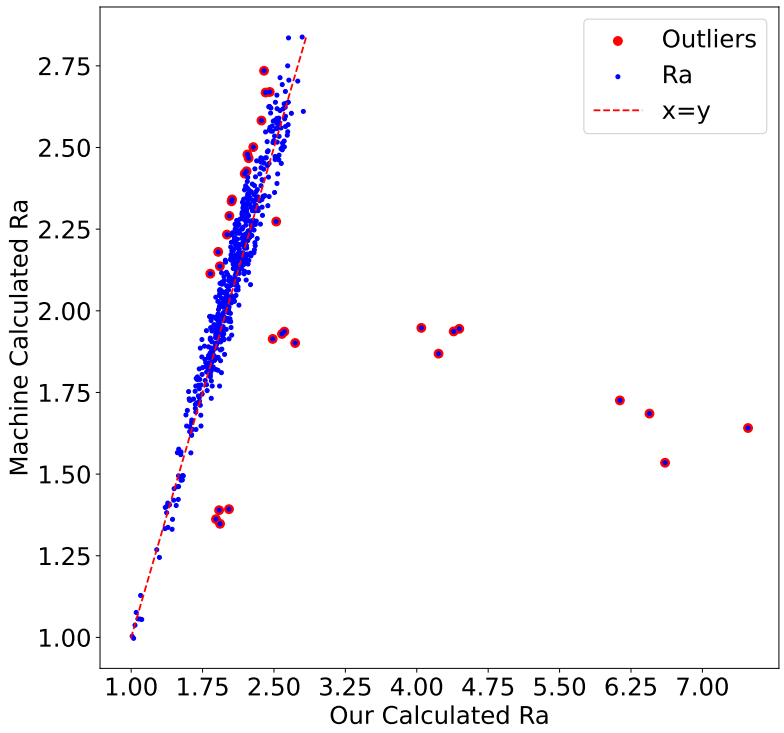


Figure 4.3: The difference between the R_a calculated from the raw laser-reflected data by us and by the measurement device itself.

been to demystify the internal workings of the device.

The critical takeaway is shown in Figure 4.4, which plots this official baseline against the stylus ground truth for each of the 49 lab samples. As quantified later in Table 8.2, this geometric baseline achieves a Pearson correlation of 0.5940. While this value suggests a moderate general trend, the visual evidence in Figure 4.4 reveals that the data is largely uninformative for production use; in the central region of the distribution, the measurements form a stochastic cluster where individual roughness values cannot be distinguished. The spread of the data points away from the trend line is too wide compared to the measurement range itself, rendering the accuracy and variability of the closed-form solution insufficient for industrial requirements.

The discrepancy between the laser-based closed-form calculation and the stylus ground truth raises the question of whether a full deterministic root cause analysis could reconcile the two measurements. At first glance, both systems measure surface geometry, suggesting that a

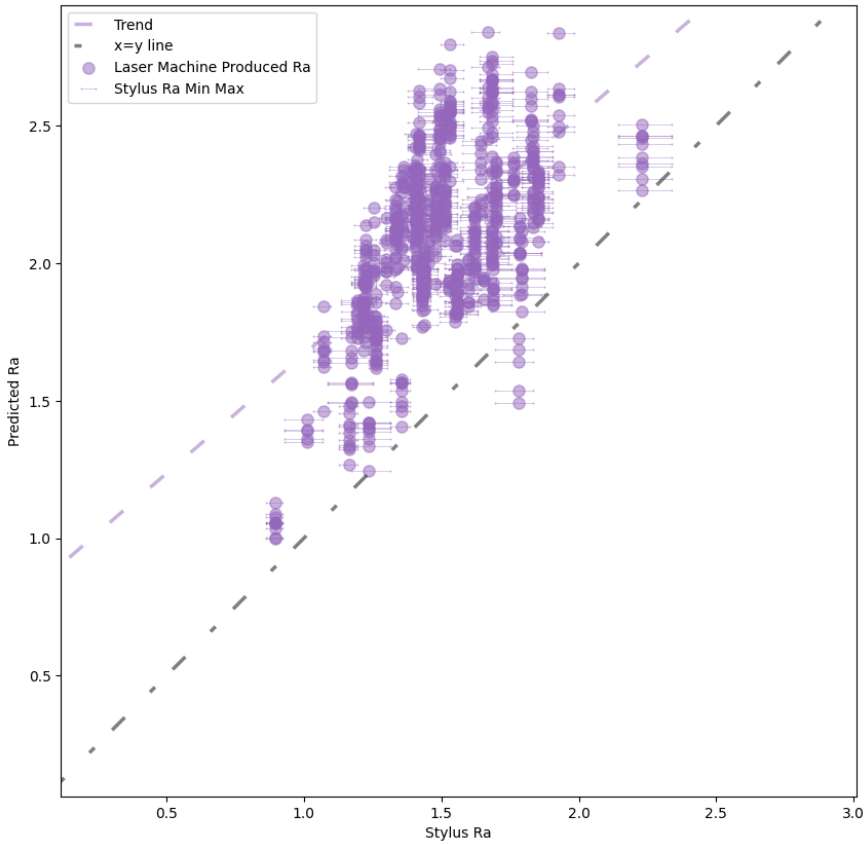


Figure 4.4: A comparison of the mean surface roughness (R_a) for each of the 49 steel samples. The x-axis value for each point is the mean of multiple R_a values produced by the laser machine. The y-axis value is the mean ground truth R_a derived from multiple stylus measurements on the same sample. The significant deviation from the ideal y=x line highlights the poor performance of the baseline method.

purely geometric explanation might be sufficient. However, several fundamental barriers limit the feasibility of such an analysis in practice.

First, the two instruments do not measure the same physical track. At the micron scale relevant for roughness analysis, even minimal spatial misalignment between the 1D laser path and the stylus trace results in effectively different surface realisations being sampled. As discussed in Section 4.3.1.1, achieving sub-micron alignment between a non-contact optical measurement and a physical stylus traversal is not practically achievable under the experimental conditions and achieving such alignment would be an extreme engineering challenge. This makes point-by-point geometric comparison ill-posed.

Second, without measurements following the same physical track, a next-best approach might be to simulate laser reflection corresponding to the same location on the surface as a

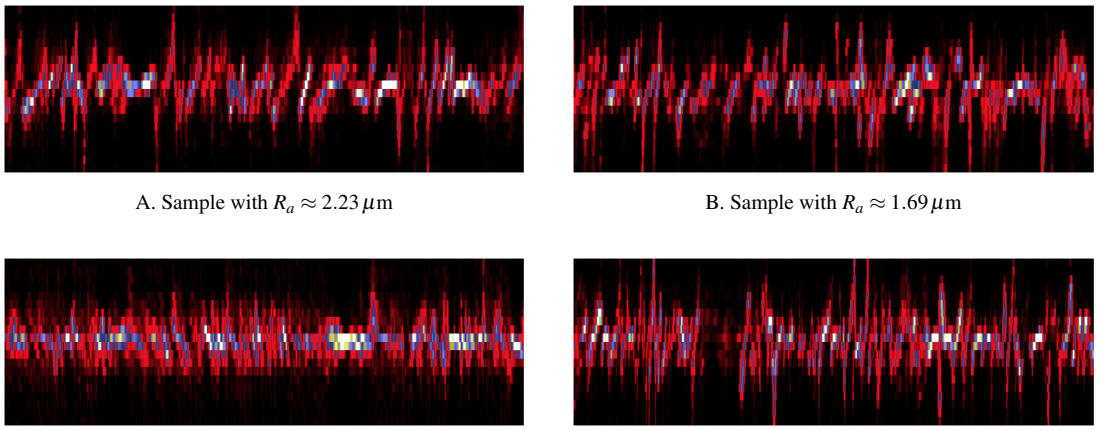
specific stylus-derived surface profile. However, this raises the question of what underlying surface profile to use to simulate the laser measurements. If a stylus-derived profile were used as the geometric basis for simulating the laser-based closed-form solution, the simulation would necessarily operate on a surface realisation different from the one that generated the recorded laser signal. Therefore, using such a simulation risks becoming an exercise in validating a mathematical model of the measurement process rather than verifying the behaviour of the real-world sensors.

Third, although both measurements are influenced by surface geometry, they arise from fundamentally different physical interactions. The stylus measures mechanical displacement, whereas the laser-based machine measures optical scattering intensity. At each timestep, the laser-based device records a 20-dimensional intensity distribution resulting from optical scattering over a small surface region. The closed-form pipeline reduces this distribution to a single slope estimate via a weighted-average angle calculation, implicitly assuming that variations in the intensity pattern are governed solely by the local surface geometry. In practice, however, optical scattering is influenced not only by slope but also by coating properties, microstructural features, reflectivity variations, and surface films. These factors may alter the angular intensity distribution independently of the true geometric slope. As a result, even if the subsequent integration and filtering steps are mathematically consistent, the reconstructed surface profile may deviate systematically from the stylus-measured geometry due to modelling assumptions embedded in the slope estimation stage.

Taken together, these constraints indicate that a purely analytical reconciliation of the two measurement modalities would require extensive physical modelling, tightly controlled alignment experiments, and hardware-level validation. Such an undertaking would constitute a substantial and distinct programme of optical metrology research.

These limitations motivate the exploration of alternative modelling strategies. Rather than relying solely on further refinements to the geometric closed-form solution of the optical signal, the subsequent chapters investigate both improvements to the existing pipeline and fully data-driven approaches. This includes structured attempts to stabilise the geometric reconstruction process, as well as models that learn the empirical mapping between laser reflection patterns and the target R_a values directly. As later results demonstrate, such approaches substantially improve predictive fidelity (approaching $R \approx 0.98$), indicating that statistical modelling provides a more effective and scalable solution for accurate on-line roughness estimation.

A key element of the novelty of this research therefore lies not in refining the closed-form



A. Sample with $R_a \approx 2.23\,\mu$m

B. Sample with $R_a \approx 1.69\,\mu$m

C. Sample with $R_a \approx 0.90\,\mu$m

D. Sample with $R_a \approx 1.26\,\mu$m

Figure 4.5: Visualisation of laser reflection data for four different steel samples, cropped to 2000 time steps for clarity. A key feature is the sparse nature of the signal, where most sensor readings are zero or near-zero (black and dark red), with high-intensity reflections appearing as intermittent bright vertical streaks. This is likely a consequence of the 1D sensor array capturing only a planar slice of the hemispherical reflection pattern.

geometric solution itself, but in demonstrating that data-driven frameworks can effectively bypass these modelling constraints and learn the empirical mapping required for high-fidelity roughness estimation in an industrial setting.

4.3 The Primary Data Sources in Detail

The research in this thesis is built upon two distinct collections of raw data, which we term the primary Data Sources. The first is a high-resolution dataset collected under controlled laboratory conditions, containing spatially-linked laser and stylus measurements. The second is a larger, more varied dataset captured during live industrial production. The following sections describe the origin, structure, and key characteristics of each.

4.3.1 The Lab Data Source

The Lab Data Source provides the high-fidelity, labelled data required for supervised learning and is derived from a total of **59** physical steel samples. These samples were collected from various industrial production batches and across multiple lines in different countries to ensure a broad and representative distribution of surface textures. The composition consists of three

coating types: 47 Galvanised (GI), 9 MagiZinc (MZ), and 3 Galvannealed (GA) [2]. From this collection, a subset of **49** samples (46 Galvanised and 3 MagiZinc) had corresponding laser reflection measurements and is therefore used for the supervised learning tasks in this thesis. The remaining 10 samples are used for the stylus-only analysis in Chapter 6. The data collection process, conducted under controlled laboratory conditions, involved taking multiple measurements of two distinct types—laser reflection and stylus profilometry—from each physical sample. Critically, these measurements were not taken at the same physical locations on the sample surface, a fact that gives rise to the significant data association challenge discussed in the later Section 4.3.1.1. The following paragraphs detail the specifics of each data component.

Table 4.1: Distribution of the number of laser scans taken per steel sample for the 49 steel samples with measurements in the Lab Data Source.

Number of samples	Number of measurements
4	5
24	10
1	23
2	24
13	25
3	26

Laser Reflection Data. The laser measurement device was used to perform multiple scans on each sample, resulting in 724 measurement files. The number of scans per sample varies between 5 and 26, as detailed in Table 4.1. Each scan has a length of $T = 65,536$ (2^{16}) measurement steps. The high sampling resolution, with a spatial interval of 0.8 µm, was achievable due to the laboratory setting, where the measurement head moved over the stationary steel samples. This sampling interval was chosen deliberately to match the standards for stylus roughness calculation. To reliably estimate surface roughness, the sampling frequency must be sufficient to capture the relevant texture features. The authoritative standard for this is ISO 3274:1996 [57], which specifies the necessary characteristics for contact stylus instruments. By choosing a laser sampling interval of 0.8 µm, it ensures the non-contact measurement resolution is commensurate with the established requirements for contact methods on such surfaces, justifying its suitability for the task.

[2] Galvanised, MagiZinc, and Galvannealed are different types of zinc-based coatings applied to steel to prevent corrosion, each with distinct surface properties.

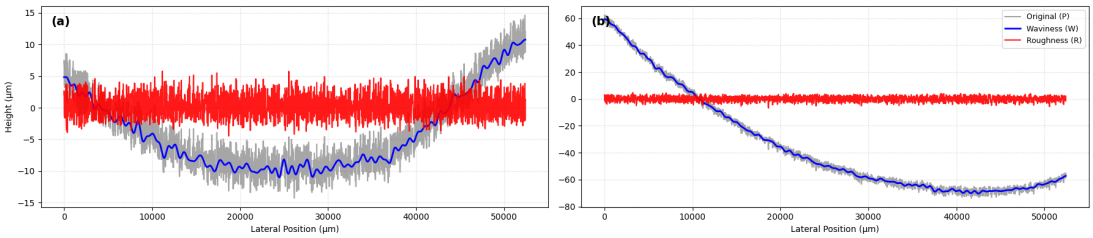


Figure 4.6: Decomposition of two representative surface profiles into their constituent waviness and roughness components. Subplot (a) shows a random profile from sample GI-034, while (b) displays a profile from sample MZ-003. In each plot, the original measured profile (P) is shown in grey. From this, the long-wavelength waviness component (W, blue solid line) is filtered, leaving the resultant short-wavelength roughness profile (R, red solid line) which is used for calculating the R_a value.

Stylus Profilometry Data (Ground Truth). The ground truth for surface roughness was obtained using a conventional stylus profilometer, with measurements collected as per the relevant ISO standards. The stylus traverses the steel surface, physically measuring height variations to create a raw surface profile. This profile is then filtered to separate the high-frequency roughness from the low-frequency waviness, as shown in Figure 4.6. The arithmetic mean roughness, R_a, is then calculated from this final roughness profile. This process is detailed in Section 2.2. Multiple stylus measurements (and therefore Ra values) were taken at different locations on each steel sample to capture its surface variability. During a manual data quality review, we discovered a small number of stylus scans contain obvious physical anomalies (such as deep scratches not representative of the overall texture) which we subsequently excluded from the dataset to ensure the integrity of the ground truth labels.

Table 4.2: Summary of Data Cleaning and Anomaly Removal

Metric	**Value**
Total number of steel samples analysed	59
Samples containing one or more anomalies	10
Percentage of samples affected by anomalies	16.95%
Initial total number of measurements	1,239
Number of anomalous measurements removed	12
Final number of valid measurements	1,227
Overall percentage of anomalies removed	**0.97%**
Percentage of anomalies (known samples only)	0.99%

A data cleaning and validation procedure needed to be performed on the initial set of stylus profiles. The raw data consisted of 1,239 measurements collected from 59 distinct steel coil

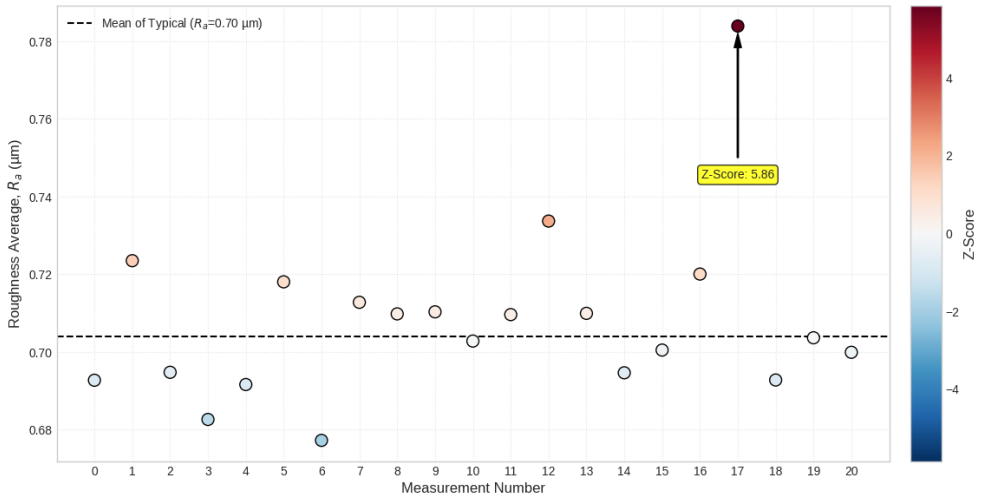


Figure 4.7: The distribution of calculated surface roughness (R_a) values for all measurements, colour-coded by their Z-score. The Z-score quantifies the deviation of each point from the mean of the typical measurements, measured in standard deviations (σ). A neutral colour indicates a Z-score close to zero, while the scale diverges to red for high positive deviations. The anomalous measurement (No. 17) is explicitly labelled with its Z-score of 5.86, identifying it as a significant statistical outlier lying nearly nine standard deviations from the mean.

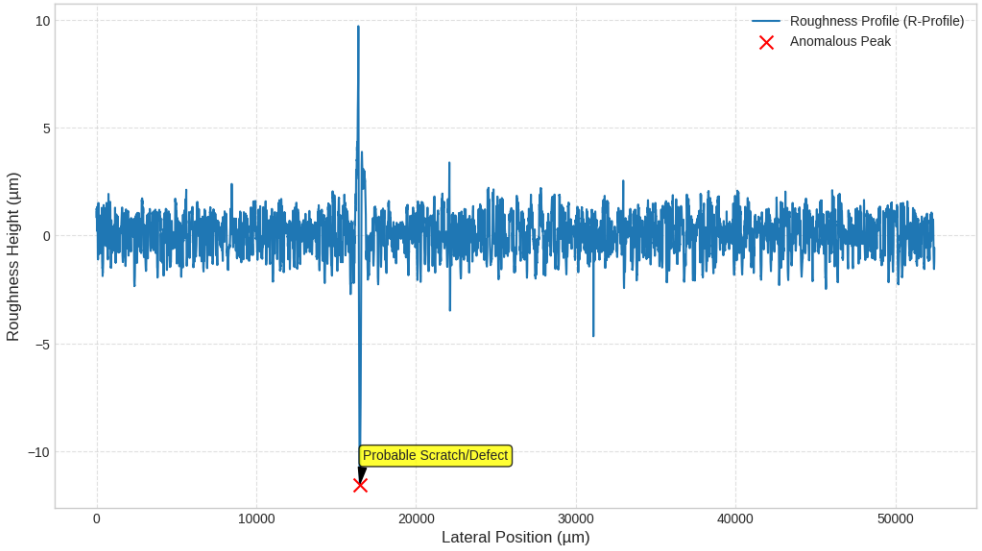


Figure 4.8: The measured roughness profile of the anomalous sample (No. 17), identified in Figure 4.7. The long-wavelength waviness component has been removed, isolating the roughness characteristics of the surface. A distinct, high-amplitude peak is visible, indicative of a physical surface defect such as a scratch or vibration. This single feature is the primary contributor to the sample's anomalously high arithmetic average roughness (R_a) value.

samples (exactly 21 each). Each measurement's roughness profile was visually inspected to identify significant anomalies inconsistent with typical surface characteristics, such as large, high-frequency peaks indicative of scratches or measurement errors. This manual screening process identified 12 measurements as outliers. These anomalous profiles were subsequently removed from the main dataset to prevent them from disproportionately influencing and causing issues.

This cleaning step is particularly important because the arithmetic mean roughness, R_a, is inherently sensitive to extreme deviations in the roughness profile. A single deep scratch or high-amplitude peak can disproportionately increase the R_a value despite not being representative of the underlying surface texture. By combining visual inspection with statistical screening (via Z-score analysis), we ensured that the final ground truth labels reflected the typical surface characteristics of each sample rather than isolated physical defects or measurement artefacts. Furthermore, the use of the mean R_a across multiple measurements per sample provides an additional stabilising effect against localised outliers.

The data removal was minimal, accounting for less than 1% of the total measurements. However, these anomalies were concentrated within a subset of the samples. As detailed in Table 4.2, 10 of the 59 steel samples (16.95%) contained at least one anomalous measurement. After their removal, a final dataset of 1,227 valid measurements was carried forward for the subsequent stages of this research. It is noted that one steel sample, labelled "unknown," contained no anomalies and its exclusion did not significantly alter the overall statistics, resulting in a final anomaly rate of 0.99% across the 58 known steel samples.

Visual and Statistical Properties A visual inspection of the data sources reveals key characteristics. The laser reflection data, visualised in Figure 4.5, appears sparse, dominated by intermittent vertical streaks of high intensity. This pattern is a consequence of the measurement physics: the 1D sensor array captures only a planar slice of the complex, hemispherical reflection pattern, with high-intensity readings occurring only when a surface facet reflects light directly into a sensor.

The stylus data, meanwhile, provides a direct physical representation of the surface. Figure 4.6 illustrates how the raw measured profile contains both the desired short-wavelength roughness and the longer-wavelength waviness, demonstrating the necessity of the filtering step. The final ground truth for model training is the arithmetic mean roughness, R_a, calculated from this filtered profile.

Quantitatively, the distribution of the ground truth R_a values for the 49-sample common set is detailed in Table 4.3. The statistics support the visual impression from the density plot (Figure 4.9), which shows a single primary mode centred near the median value of 1.48 µm. The distribution is not symmetric, however, exhibiting a slight positive skew (skewness = +0.19). This is visible in the plot as a wider tail on the right-hand side, indicating a sub-population of samples with higher roughness values. This asymmetry is likely due to the presence of distinct steel processing batches or different coating types within the sample set.

Table 4.3: Descriptive statistics for the ground truth ‘Stylus Ra‘ values across the 49 common samples.

Statistic	**Value (µm)**
Mean	1.4919
Median	1.4802
Standard Deviation	0.2603
Minimum	0.8971
Maximum	2.2287
Skewness	0.1923

4.3.1.1 The Many-to-Many Labeling Problem and Solution

A significant challenge in creating this dataset is the spatial misalignment between the laser and stylus measurements. Due to the microscopic scale, it is not feasible to ensure that the laser scanner and the stylus measure the exact same track on the steel surface. This creates a "many-to-many" correspondence problem, as illustrated in Figure 4.10: each steel sample has multiple laser scans and multiple stylus measurements, with no direct one-to-one mapping between them.

To resolve this and to facilitate supervised learning in the later chapters, we adopt a sample-level labelling strategy. For each of the 49 steel samples, we calculate the mean R_a across all of its stylus measurements. This single mean R_a value is then assigned as the ground truth label to all laser scans from that same steel sample. This approach is a necessary simplification that introduces a degree of label noise, as it assumes the surface roughness is reasonably homogeneous across a given sample. However, it is a justifiable strategy for two reasons. First, the averaging process produces a robust estimate of the sample's central tendency for roughness. Second, a robust machine learning model should be able to learn the underlying relationship between laser reflection patterns and this mean roughness value, effectively treating the intra-sample variation as noise to be learned from.

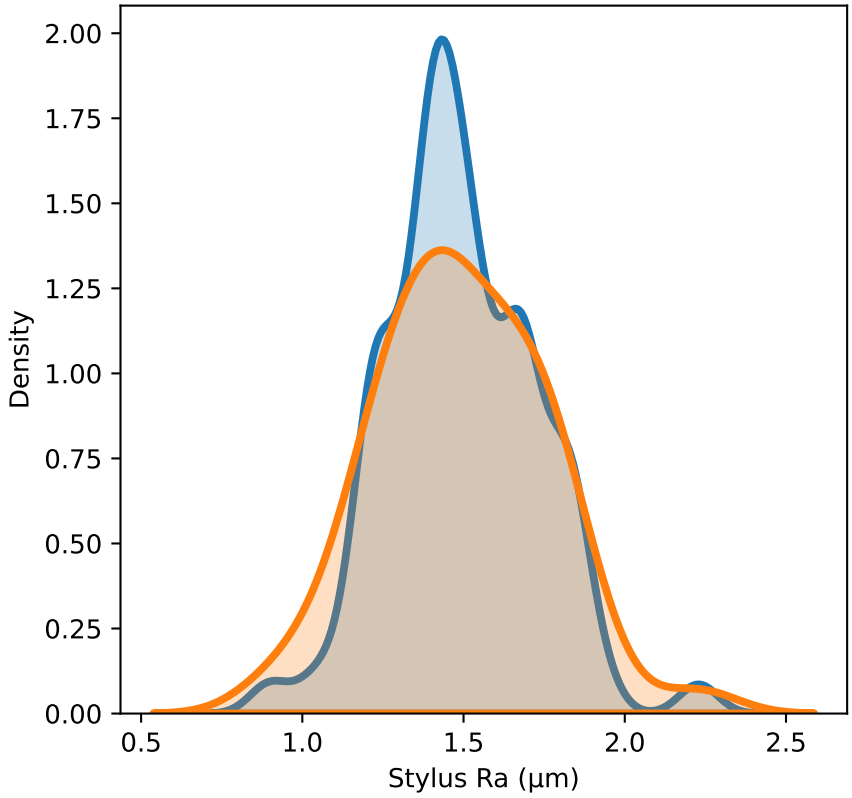


Figure 4.9: The probability density distribution of the arithmetic mean roughness (R_a) from the stylus measurements (filtered to contain only those steel samples with corresponding laser measurements). The blue curve shows the distribution of all individual stylus measurements. The orange curve shows the distribution of the mean R_a values, where each of the 49 steel samples contributes one value.

4.3.2 The In-Production Data Source

A second, much larger dataset comes directly from the production line. This dataset contains only on-line laser measurements with no related stylus measurements taken. It has 52,388 laser scans from 112 different steel coils. Each scan covers approximately 5 cm of surface length with a sequence length of $T = 50,000$. The sampling interval of 1.0 μm is larger than in the lab data, as this is the minimum achievable resolution for the device when operating on the live production-line where the steel strip is in motion. Due to the lack of any stylus measurements this data is missing anything which might be used as ground truth labels. However, this data source is significantly larger and captures a broader diversity of surface conditions, making it a beneficial resource for additional experiments. Nevertheless, this in-production data is subject to additional real-world noise sources not present in the lab, such as varying levels

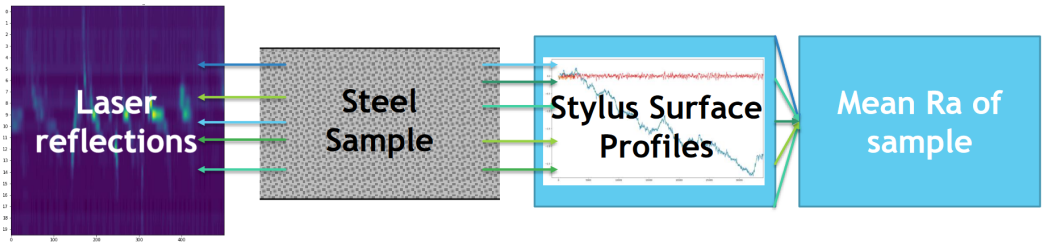


Figure 4.10: Diagram illustrating the many-to-many data problem. Multiple laser scans and multiple stylus measurements are taken from different locations on the same steel sample. A single mean R_a value, derived from all stylus profiles of a sample, is used as the target label for all laser scans from that sample.

Table 4.4: A summary of the key statistics for the labelled and unlabelled datasets.

Property	Labelled Dataset	Unlabelled Dataset
Data Source	Lab Samples	In-Production
Number of Steel Samples/Coils	49	112
Total Number of Laser Scans	724	52,388
Surface Roughness (R_a) Labels	Available (Mean per sample)	Not Available
Coil/Sample ID	Available	Available
Sequence Length (T)	65,536	50,000
Sampling Interval	0.8 μm	1.0 μm
Physical Scan Length	$\approx$ 5.24 cm	$\approx$ 5.00 cm

of lens fouling, intermittent 'hot' sensor readings from electrical noise, and other transient production-line defects. While this noise presents data issues, it is a realistic reflection of the data that can be gathered in the real-world, allowing for pre-training of more robust models.

4.3.3 Summary

In this chapter, we have introduced the data collection procedures and detailed the two primary **Data Sources** that form the basis of our experiments. We have described the composition of the *Lab Data Source*, used for supervised learning, and the larger, more challenging *In-Production Data Source*, suitable for self-supervised pre-training. We have also detailed the fundamental challenges inherent in this data, such as the many-to-many labelling problem, caused by the misalignment between the locations of laser and ground truth measurements, and real-world noise. These inherent characteristics and limitations of the available data, coupled with the performance gaps identified in the existing literature (Chapter 3), motivate the specific research problems and questions formalised in the following chapter. Having established the nature of

the data and the significant limitations of the existing physics-based baseline, the subsequent chapters will explore how machine learning can be leveraged to address these challenges.

Firstly, we investigate how imputation approaches can improve the data quality of our dataset of stylus measurements, allowing for higher accuracy in the closed-form calculation of the R_a value from this data. Later, in Chapter 8, we evaluate various machine learning approaches for their accuracy in predicting R_a values, making use of our smaller, labelled dataset for training. In Chapter 9 we then expand our approaches to include the larger, unlabelled dataset for unsupervised pretraining. The results of the experiments in all chapters demonstrate the potential for machine learning to improve accuracy in the prediction of values imperative to steel production.

Chapter 5

Problem Statement, Hypotheses, and Contributions

Over the course of the preceding chapters, we have introduced context about the industrial domain, steelmaking, where the contributions of this thesis are made, as well as laying the foundation of machine learning required to tackle problems within this domain. In the Data Chapter (Chapter 4) we have also detailed the data provided by our industrial partner in order to perform our research. Building upon the review of related work in Chapter 3 and the detailed analysis of the specific data sources, acquisition challenges, and baseline performance presented in Chapter 4, this chapter bring together and formalises the core research problems addressed in this thesis. Let us first review the major problems within modern steelmaking which we aim to address.

Accurate surface roughness quantification is a cornerstone of high-value steel manufacturing, where surface characteristics directly impact product performance and customer satisfaction. In particular, the arithmetic mean roughness parameter, R_a, is the primary industry-standard metric, contractually specified for automotive panels and other critical steel products. The prevailing workflow relies on stylus-based profilometry, which, while precise, suffers from significant limitations: measurements are labour-intensive, spatially sparse, and delayed, hindering real-time process optimisation and resulting in costly reworks when specifications are not met. This measurement delay is particularly disruptive for just-in-time manufacturing workflows, introducing significant bottlenecks and inefficiencies into the supply chain. Furthermore, the inherent lack of full coil coverage from these sparse measurements represents a missed opportunity for comprehensive quality assurance, which could otherwise serve as a

value-added selling point for customers or be used for detailed internal analysis.

To address these industrial challenges, laser-based on-line sensing systems have been developed to enable continuous, non-contact measurement of surface properties as steel coils are produced. However, for the type of system used in this research, the specific challenge of translating raw laser reflection data into reliable surface roughness statistics remains a difficult scientific and engineering problem. The physics underlying laser scattering from steel surfaces is complex and is not described with high fidelity by closed-form models, particularly when real-world noise, material variations, and sensor imperfections are present. Direct predictions by laser measurement devices routinely diverge from stylus ground truth, undermining confidence in their results and limiting their adoption as robust process control tools.

In particular, we hypothesise that the laser measurement device used by our industrial partner has a fundamental limitation in its inability to decouple the surface gradient from the sensor's focal distance. Therefore, variations in the steel's vertical position can be misinterpreted as changes in surface texture, introducing a source of measurement error.

Building on the foundational material provided earlier, this chapter defines the scope of the thesis and articulates its central research question:

> *How can machine learning models be designed, trained, and validated to reliably predict R_a from imperfect, high-volume on-line laser data, overcoming the limitations of both physical modelling and sparse and poor ground-truth labelling?*

In the following sections, we first formalise the specific challenges we aim to address in this thesis, before outlining our hypotheses. Finally, we provide a detailed summary of our contributions.

5.1 Problem Statement

Although the contributions of this thesis are all designed to address the central research question, specifically enabling the prediction of the surface roughness parameter, R_a, there are multiple angles from which we approach this challenge. In this section, we will outline each of the problems individually, explaining why they are novel and not easily solved by existing research, as well as introducing four specific research questions to address them.

Imputing Missing Data Gaps in Steel Surface Profiles First, we address the problem of restoring poor-quality steel surface data by imputing gaps where measurements are poor

enough to be considered missing gaps. One hypothesis is that this problem is caused by the fact that the laser measurement head uses a 1D array of sensors to capture intensities of laser light reflected from the steel surface. Since the surface is three-dimensional and reflections scatter in many directions, we believe that this could lead to sparse signals when portions of the laser light scatter outside the 1D sensor plane. If we are able to effectively identify these poor regions and impute them with more realistic data, the closed-form physics-based calculation of the R_a value may demonstrate improved accuracy. This challenge is the focus of Chapter 6, and in Section 6.2 we explore further the prevalence of this issue. A paper is in preparation based on the work of this chapter.

Existing solutions, such as simple interpolation methods or even deep learning imputation models, are not specifically designed for, and therefore not optimal for, our setting. In particular, this is because most existing approaches optimise for pointwise accuracy, which does not directly address our need to preserve the R_a value of the steel sample as a whole. In some cases, good pointwise accuracy can result in a very inaccurate value for R_a when compared to the ground truth. Thus, it is necessary to design solutions specific to our problem. Our research question to this end is:

> *R1: How can missing data gaps in steel surface profiles be effectively imputed such that reconstructed data accurately preserves key roughness statistics required to support reliable surface characterisation?*

Improving the ROCKET Model for Surface Roughness Prediction The second problem we address in this thesis centres on improving the accuracy and efficiency of the ROCKET model, particularly with our goal of applying it to R_a prediction in mind. ROCKET (Random COnvolutional KErnel Transform) is a recent and computationally efficient framework for time series classification and regression. In our initial experiments, we identified ROCKET as a promising approach for predicting surface roughness directly from sensor data. However, the model suffers from computational limitations, presenting challenges for real-time applications such as ours, and we further hypothesise that its accuracy could be improved with more sophisticated pruning strategies.

Though existing work has shown promising performance of the ROCKET model, we present explorations of novel methodologies for improving it. In the work of Chapter 7, we address the following research question:

R2: How can ROCKET-based models be optimised for the industrial prediction of steel surface roughness, firstly by improving computational throughput via GPU implementation, and can efficiency improvements allow fine-tuning to improve predictive accuracy with better model efficiency?

Prediction of the R_a Surface Roughness Parameter The third problem tackled in this thesis directly addresses our central research question. Chapter 8 is concerned with the robust and accurate prediction of the R_a roughness parameter from laser sensor data using machine learning models. This approach is a more direct way to apply ML to improve R_a predictions than imputing data gaps to improve the closed-form calculation. The work of this chapter was published in the Springer International Journal of Advanced Manufacturing Technology [85].

While the prediction of surface roughness using machine learning has been explored in other domains, that body of work is situated in manufacturing contexts fundamentally different from high-speed steel rolling. The majority of studies focus on processes such as machining, milling, or additive manufacturing [68]. In these settings, the typical approach is to predict the final surface roughness from a set of known process parameters (e.g., tool speed, laser power) or from auxiliary sensor data like tool vibration [37]. By contrast, the work in this chapter addresses the novel challenge of learning a direct transformation from high-frequency, on-line optical sensor data to the stylus-equivalent R_a value. To the best of our knowledge, this is the first comprehensive comparative study of diverse deep learning architectures for this type of industrial problem, using real-world data from a continuous production line.

This work also contrasts with the work of Chapter 6, where imputation of missing data improves the closed-form calculation of R_a from the laser measurement data. Here, we train machine learning models with ground truth stylus measurements, more closely aligning predictions with the gold standard.

An unprecedented empirical evaluation of ML methods on authentic on-line measurement signals is enabled by our large-scale dataset collected from an industrial temper roll steel production line, which contains raw laser sensor readings and a partially related stylus-measured R_a ground truth labels. A key challenge, which this research addresses, is that the link between the optical signal and the ground truth is complex and indirect, rather than a simple one-to-one mapping. Thus, our research fills a significant gap in understanding which modelling approaches are most accurate for R_a estimation from noisy, real-world laser reflection data. In particular, there are competing paradigms which may work well for our data, as it can be naturally expressed using either a 1D or 2D format, motivating the use of both 2D vision

models and state-of-the-art 1D time series models. We compare these against each other, as well as non-deep learning approaches in a fully comprehensive study. We address the following research question:

> *R3: Which machine learning model architectures and techniques provide accurate and robust predictions of the R_a roughness parameter from noisy, high-volume, industrial laser sensor data?*

Leveraging Pretraining for Improved R_a Prediction The final challenge addressed in this thesis is the question of how pretraining techniques such as Self-Supervised Learning (SSL) can be adapted in order to improve R_a prediction. Representation learning and SSL have been extensively studied in the fields of computer vision and time-series analysis. Moreover, they have seen successful application in manufacturing and materials science contexts for extracting informative features from data without exhaustive manual labelling.

However, the unique characteristics of the data arising from steel surface monitoring, especially the high-aspect-ratio spatial arrays formed by laser reflection sensor arrays, pose distinctive challenges. Existing representation learning methods have not been applied to these complex data modalities, and their potential role in enhancing surface roughness regression for industrial steel production remains unexplored. We present novel research which adapts existing pretraining approaches for our setting in the work of Chapter 9, which is based on work which has been accepted to the Springer conference Advanced Concepts for Intelligent Vision Systems (ACIVS) 2025 [86]. We are inspired by the work of the previous chapter in our choice of backbone architecture, drawing from our empirical evaluation to determine which model architecture is most effective for our specific industrial data. The research question guiding this work is:

> *R4: Can representation learning using large-scale unlabelled production sensor data improve the accuracy, robustness, and generalisation of machine learning models for Ra surface roughness prediction?*

5.2 Hypotheses

Having introduced the research questions that guide our work, we now define two central hypotheses. Note that we do not begin with preconceptions about which machine learning

approaches will be best suited for our data, a key philosophy which motivates such a comprehensive study of techniques. Therefore, we do not hypothesise about the relative performance of specific models.

- **H1:** In answer to RQ1, we hypothesise that a deep learning model trained with a custom, statistics-aware loss function that directly penalises deviations in R_a will outperform both traditional interpolation methods and standard deep learning models (trained on MSE alone) in preserving the statistical integrity of the reconstructed surface profile.
- **H2:** In answer to RQ3, we hypothesise that at least some ML approaches are able to outperform the closed-form physics-based calculation performed by the SORM machine.
- **H3:** In answer to RQ4, we hypothesise that pretraining on large-scale unlabelled production data via representation learning will enhance supervised learning performance for R_a prediction, yielding models with improved accuracy
- **H4:** In answer to all of RQ2, RQ3 and RQ4, we hypothesise that ML models specifically adapted or optimised for the unique characteristics of steel surface sensor data, such as through tailored pruning, feature engineering, custom loss functions or architectural innovations, will outperform generic unadapted ML approaches in predicting R_a.

The upcoming chapters of the thesis will perform experiments in pursuit of answering our proposed research questions, and proving or disproving the listed hypotheses.

5.3 Summary of Contributions

To conclude this chapter, we present a list of the contributions made in the upcoming chapters. Each of these contributions directly progresses the research in aid of answering our research questions:

- The development of a methodology for identifying poor-quality data segments in laser reflection measurements, drawing on a combination of three quality metrics
- A study evaluating the performance of contrasting types of machine learning model for the imputation problem arising from missing data, including the incorporation of a novel statistics-based loss

- An efficient GPU-accelerated implementation of the previously proposed ROCKET model
- An experimental evaluation of fine-tuning approaches on a pruned set of kernels to assess the impact on ROCKET's accuracy and efficiency
- A thorough comparative study of various types of machine learning approach to predict the R_a roughness parameter of steel surfaces directly from high aspect ratio input data, with implications for other similarly high aspect ratio datasets
- The evaluation of multiple pretraining strategies that leverage unlabelled data for better representation learning
- Included in this evaluation is a novel autoencoder architecture which incorporates a TCN as both the encoder and decoder
- Analysis of experimental results to identify several machine learning models that demonstrate improvements over baseline approaches, and optimal strategies for predictive accuracy in this domain problem

In this chapter, we have established the problems we aim to tackle within this thesis, including the formulated research questions, hypotheses and final contributions. We will now transition to addressing these problems in the next chapter, where we will begin with the first of these: the imputation of missing surface profile data.

Chapter 6

Reconstruction of Surface Profiles for Enhanced Roughness Estimation

6.1 Introduction and Motivation

While rich in information, the on-line laser reflection data exhibits characteristics suggestive of localised data loss. As detailed in Chapter 4, the measurement head captures a 1D arc of light reflected from the steel surface. Given the inherently 3D nature of surface topography, along with our industrial partner, we hypothesise that portions of the laser light can be scattered outside this 1D plane, leading to time steps where the recorded sensor intensities are anomalously low or highly scattered. We interpret these phenomena as segments of "missing" or unreliably captured surface information. Unaddressed, such data gaps could impact the accuracy of downstream analyses, particularly the calculation of critical surface roughness parameters like R_a.

The established closed-form solution for deriving an estimated surface profile from this laser data relies on integrating gradients of the reflections. Such an approach is inherently sensitive to these low-signal or highly scattered regions. If these problematic laser data segments are used directly, they could lead to erroneously interpreted sections in the calculated profile, thereby degrading subsequent R_a calculations. A potential strategy within a pipeline utilising the laser data would be to first identify these unreliable segments (using metrics such as those developed later in Section 6.2) and then, instead of using the flawed laser data to inform the profile in these regions, intentionally treat these corresponding sections of the calculated estimate of the surface profile as missing data requiring imputation.

However, evaluating the efficacy of gap-filling techniques directly on such a "laser-derived profile with gaps" presents significant challenges. The baseline profile itself is an estimate, potentially containing inherent noise and errors from the closed-form solution, which makes it difficult to isolate and accurately assess the performance of the reconstruction algorithm alone. Therefore, to robustly develop and rigorously evaluate methods for reconstructing missing segments, this study focuses on a more controlled problem designed to isolate the core challenge of reconstructing the high-frequency surface texture that dictates roughness.

This is achieved by formulating a controlled experimental framework where we utilise high-fidelity stylus profilometry data. This data represents a true and known surface profile where we introduce artificial, controlled gaps. The primary goal is to develop and evaluate methods for reconstructing these missing segments in such a way that the statistical integrity of the profile, especially concerning roughness parameters, is preserved or improved. By systematically assessing different techniques on this controlled data, we can rigorously quantify the performance of the gap-filling methods themselves. The insights and validated techniques gained from this investigation are intended to be ultimately transferable, informing strategies to effectively fill gaps in profiles derived from the more complex laser data pipeline. As illustrated in Figure 6.1, the models developed and validated within this controlled study are designed to serve as a robust component in a larger system, ultimately improving R_a prediction and the quality of profiles obtained from traditional closed-form solutions.

The complete methodology, from development to its intended application, is illustrated conceptually in Figure 6.1. The work presented in this chapter corresponds entirely to the development phase (Stage 1). This phase involves two parallel efforts. First, we conduct a detailed analysis of the real-world laser data to develop and validate a robust function for identifying poor-quality segments (Stage 1A). Second, using high-fidelity stylus profilometry data as a ground truth, we establish a framework for training and evaluating deep learning imputation models. Critically, the statistical properties of the gaps discovered in Stage 1A are used to guide the simulation of artificial gaps within the stylus data, ensuring our evaluation is representative of the real-world problem (Stage 1B). While outside the scope of the current chapter, a future study could integrate these components into the existing pipeline with the aim of significantly improving the quality of the final surface roughness statistics derived from on-line sensor measurements. This could enable the application pipeline shown in Stage 2. At the current time, a paper is in preparation based on the work of this chapter.

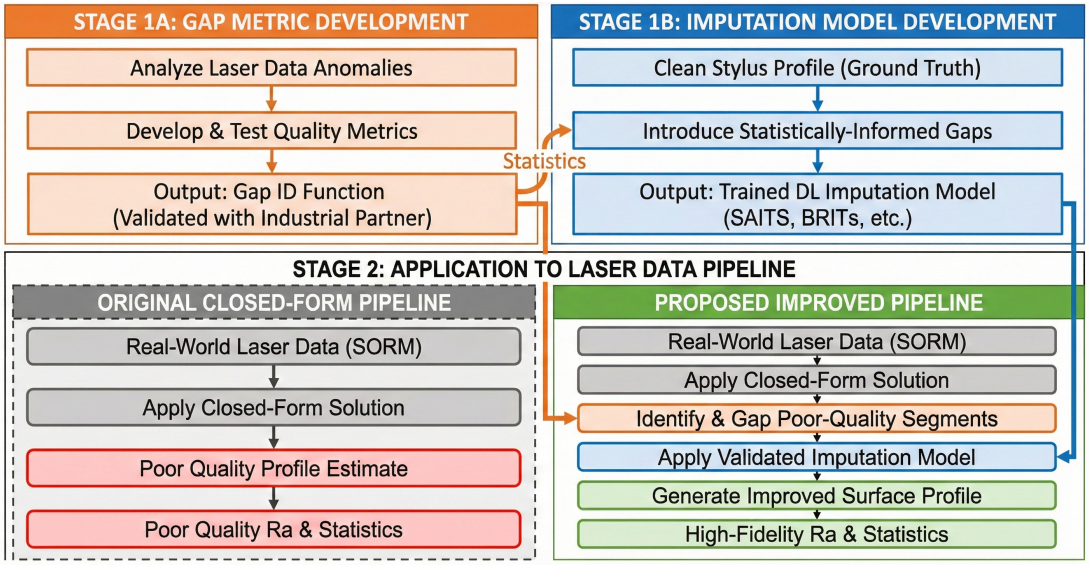


Figure 6.1: Conceptual diagram of the two-stage research methodology. Stage 1 is the development phase, where a Gap ID Function is created from laser data (1A) and a DL Imputation Model is trained on stylus data with statistically-informed controlled gaps (1B). Stage 2 contrasts the original closed-form pipeline with the proposed application, demonstrating how these two validated components are integrated to produce higher-fidelity surface statistics.

6.2 Characterising Potential Data Gaps in Laser Reflection Profiles

A formal, automated methodology is required to quantify and characterise suspected data quality issues within the laser reflection measurements. The dataset, as detailed in Chapter 4, provides the basis for this analysis. Visual inspection of the data reveals significant inconsistencies, including signal dropouts and variable scattering patterns, suggesting that data quality is not uniform. The design of a method to automatically identify these issues is motivated by several hypothesised physical causes for poor data quality, including: incorrect distance between the sensor array and the steel surface leading to a loss of laser focus; out-of-plane reflection causing the beam to miss the sensor array entirely; and complex light-trapping interactions on very rough surfaces, where light is scattered multiple times between peaks and valleys instead of reflecting directly back to the sensors. The metrics developed are validated through collaboration with our industrial partner, whose domain experts visually confirm that the regions ultimately flagged by our methodology are representative of what they consider to be poor-quality data signals.

6.2.1 Methods: Development of Quality Metrics

To identify poor-quality data segments, a robust, multi-metric approach was developed. This final methodology combines three distinct metrics, each designed to capture a different facet of data quality. A timestep is flagged as "poor quality" if its score falls below the chosen threshold for any of the three metrics. Figure 6.2 provides a detailed visualisation of how these individual metrics contribute to the final flagging decision on a representative sample.

The three metrics are described as follows:

1. **Log-Normalised Intensity Score (S_I):** This metric measures the overall strength of the signal to flag periods of significant light loss. For each timestep t, the peak sensor index i_{peak} is identified. The raw intensity of the peak cluster, $I_{raw}(t)$, is calculated by summing the absolute sensor values within a defined cluster width ($w_c = 1$) around the peak: $I_{raw}(t) = \sum_{i=i_{peak}-w_c}^{i_{peak}+w_c} |S_i(t)|$. To handle the large dynamic range of these values, a logarithmic compression is applied: $I_{log}(t) = \log(1 + I_{raw}(t))$. Finally, this log-compressed value is min-max normalised across all timesteps in the sample to produce the final score, $S_I(t)$, in the range [0, 1].

2. **Global Variation Score (S_V):** This metric measures the "peakiness" or "spikiness" of the signal across all sensors, distinguishing a focused reflection from uniform noise or a signal plateau. For each timestep t, the standard deviation of all 20 sensor values, $\sigma(t) = \text{std}(S_1(t), \ldots, S_{20}(t))$, is calculated. This array of standard deviations is then min-max normalised across the entire sample to a linear score, $V_{linear}(t)$, in the range [0, 1]. To better distinguish between different levels of low variation, a gamma correction is applied to produce the final score: $S_V(t) = (V_{linear}(t))^{\gamma}$, where γ was set to 0.75.

3. **Focus Percentage Score (S_F):** This metric measures the concentration of energy within the peak cluster relative to the total energy in that timestep. To handle standardised data robustly, the entire sample is first min-max normalised to the range [0, 1]. For each timestep t, the score is then calculated as the ratio of the energy in the peak cluster (defined here with $w_c = 1$) to the total energy of that timestep. The final score is given by the equation:

$$S_F(t) = \frac{\sum_{i=i_{peak}-w_c}^{i_{peak}+w_c} S'_i(t)}{\sum_{j=1}^{20} S'_j(t)} \tag{6.1}$$

   where S' represents the min-max normalised sensor values for that timestep.

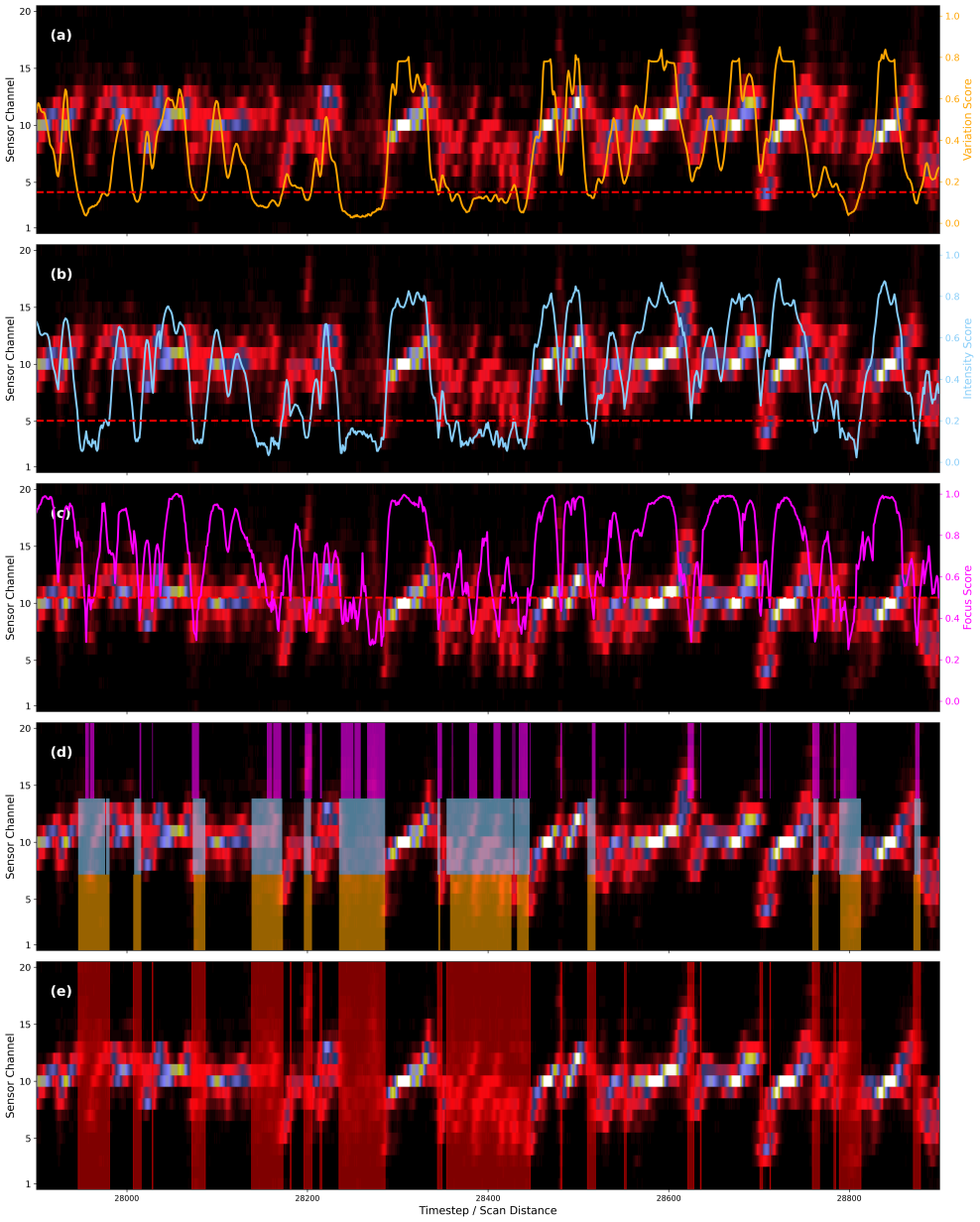


Figure 6.2: A detailed breakdown of the combined metric flagging process on a representative "typical" sample, steel sample GI031, which has 27.33% of its data flagged as poor quality. **(a-c)** The top three panels show the response of the individual metrics (Global Variation, Log-Normalised Intensity, and Focus Percentage, respectively). The red dashed line indicates the threshold below which a timestep is flagged by that specific metric. **(d)** The fourth panel visualises the overlaps of these individual flags, with each metric's flagged region assigned its own colored horizontal lane. **(e)** The bottom panel shows the final combined result, where a timestep is highlighted in red if any of the three component metrics fall below their respective thresholds.

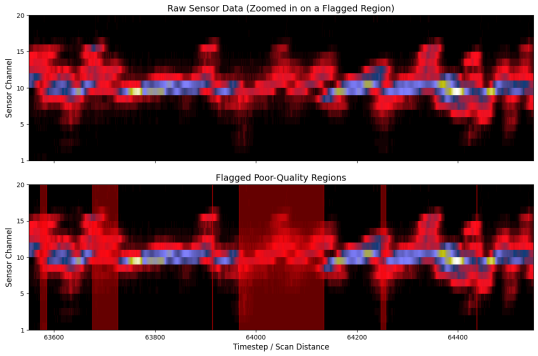
It should be noted that this combination of metrics was chosen after an iterative development process. For example, Shannon Entropy was initially evaluated as a potential measure of signal focus but was ultimately discarded. Its "scale-invariant" nature proved problematic, as it failed to distinguish between low-energy, uniform noise (present in dark frames) and high-energy, uniform signals (such as a wide plateau from a specular reflection), both of which can produce high entropy values. The final three metrics, however, provided a much more robust and reliable system for identifying poor-quality data.

Based on empirical evaluation and validation with our industrial partner, the final thresholds for flagging a timestep as poor quality are set as: a **Variation Score (S_V) < 0.15**, an **Intensity Score (S_I) < 0.2**, or a **Focus Score (S_F) < 0.5**. As a final post-processing step, we merge any small, isolated non-flagged segments of 2 timesteps or less that are surrounded by flagged regions. This operation makes the identified poor-quality segments more contiguous, which is more representative of a continuous signal dropout event.

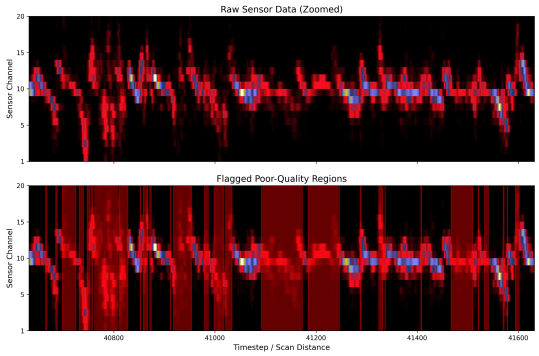
6.2.2 Quantitative Analysis of Data Quality Based on Chosen Metrics

Applying this methodology to the entire dataset of 775 samples provides a comprehensive, quantitative overview of the data quality issues. Across the entire dataset, the analysis reveals 28.57% of all timesteps are flagged as poor quality. This issue, however, is not uniformly distributed. The percentage of poor-quality data within a single sample varies dramatically, ranging from a minimum of 14.2% to a maximum of 96.2%, with a standard deviation of 9.7% around the mean of 28.57%. This variability is consistent across the predefined training, testing, and validation splits.

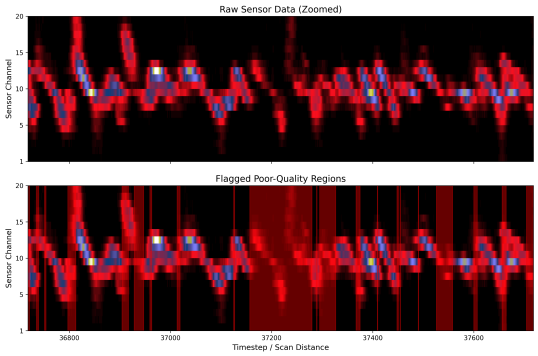
The analysis strongly correlates poor data quality with specific steel types, as summarised in Table 6.1. The non-galvanised steel types, identified as ‘MZ001’, ‘MZ002’, and ‘MZ003’, consistently exhibit the highest average percentages of compromised data. This finding is consistent with the known physical properties of the surface coating on these samples. The ‘MZ’ identifier corresponds to a MagiZinc coating, a modern zinc-aluminum-magnesium alloy system. Unlike traditional hot-dip galvanising, which can result in a highly specular surface with large crystalline “spangles”, these advanced coatings are engineered to have a "spangle-free smooth aspect” [6]. A spangle-free, matte surface is inherently a diffuse scatterer; therefore, it is expected that this coating would cause a significant portion of the incident laser light to scatter away from the 1D sensor array, leading to the reduced signal intensity and focus that our metrics are designed to detect. This is visually demonstrated in Figure 6.3, which contrasts



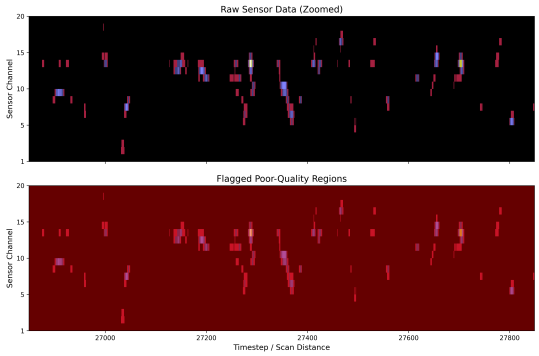
A. Best Case (0th Percentile) (Steel Sample: `GI023`), with 14.18% of data flagged as poor quality.



B. Good Case (25th Percentile) (Steel Sample: `GI033`), with 24.00% of data flagged as poor quality.



C. Problematic Case (75th Percentile) (Steel Sample: `GI046`), with 31.44% of data flagged as poor quality.



D. Worst Case (100th Percentile) (Steel Sample: `MZ003`), with 96.18% of data flagged as poor quality.

Figure 6.3: Visual comparison of data quality for four representative samples across the dataset. Each example shows the raw sensor data (top) and the same data with timesteps flagged as poor quality highlighted in red (bottom). The examples are chosen to represent the best, 25th percentile, 75th percentile, and worst cases based on the percentage of flagged data. The displayed segments are zoomed excerpts selected to illustrate missing regions; percentages refer to the full signal. Consequently, the apparent density of flagged regions in the zoomed plots does not necessarily reflect the global percentage of flagged data.

samples from across the quality spectrum.

The distribution of the lengths of these contiguous poor-quality segments is also analyzed, as shown in Figure 6.4. The distribution is heavily right-skewed, with a very high frequency of short gaps (the majority being less than 20 timesteps). However, a long tail of much larger gaps, extending beyond 100 timesteps, is also a notable feature of the data.

The quantitative findings from this work provide the empirical basis for the problem of data imputation explored in the subsequent section. The characteristics identified are used to parameterise the artificial generation of gaps in the high-fidelity stylus data, ensuring that the

Table 6.1: Comparison of steel sample types with the highest (worst) and lowest (best) average percentage of poor-quality data.

Steel ID	Mean Poor Quality (%)	Std Dev (%)	Min (%)	Max (%)
Highest Poor Quality (Worst Signal Integrity)				
MZ003	95.38	1.13	93.42	96.18
MZ002	83.85	3.22	80.49	88.42
MZ001	72.52	1.09	71.00	74.04
GI047	60.03	1.36	58.24	61.42
GI013	38.03	0.94	36.13	39.32
Lowest Poor Quality (Best Signal Integrity)				
GI023	16.28	1.08	14.18	17.98
GI001	18.56	1.01	17.22	20.23
GI028	19.01	1.15	16.81	20.85
GI005	19.26	1.22	17.89	21.40
GI042	20.68	1.04	19.16	22.03

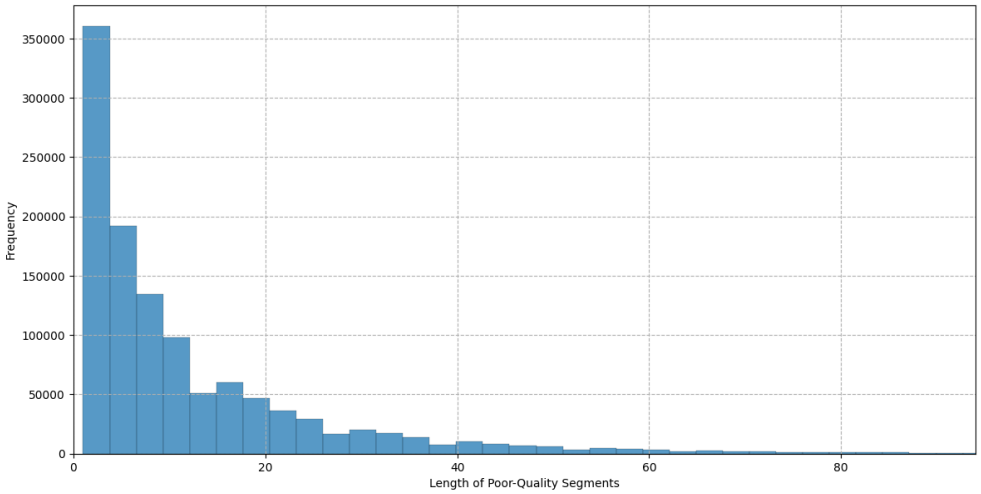


Figure 6.4: Distribution of the lengths of contiguous poor-quality segments identified across the entire dataset. The y-axis is the Frequency, highlighting the high prevalence of shorter gaps while also showing the existence of a long tail of larger gaps.

controlled experiments on data reconstruction are designed to reflect the scale and nature of the challenges present in the real-world sensor data.

6.3 Methodology: Reconstructing Gaps in Stylus Roughness Profiles

In order to achieve a framework for testing, we formulate a controlled problem using true stylus profilometry data where controlled, artificial gaps can be introduced to precisely measure reconstruction performance against a known true section.

As established in Chapter 4, a direct one-to-one mapping between a laser measurement and a stylus measurement is not available, and a height profile derived from laser data contains its own estimation errors. Therefore, to isolate the performance of the gap-filling techniques themselves, high-fidelity stylus data is used as a known ground truth. Furthermore, to ensure the models focus on the most critical part of the signal, we perform our experiments on the roughness profile. The true scientific challenge lies not in reconstructing low-frequency waviness (the trend), but in reconstructing the complex, high-frequency roughness component that determines the R_a statistic.

6.3.1 Data Preparation and Gap Simulation

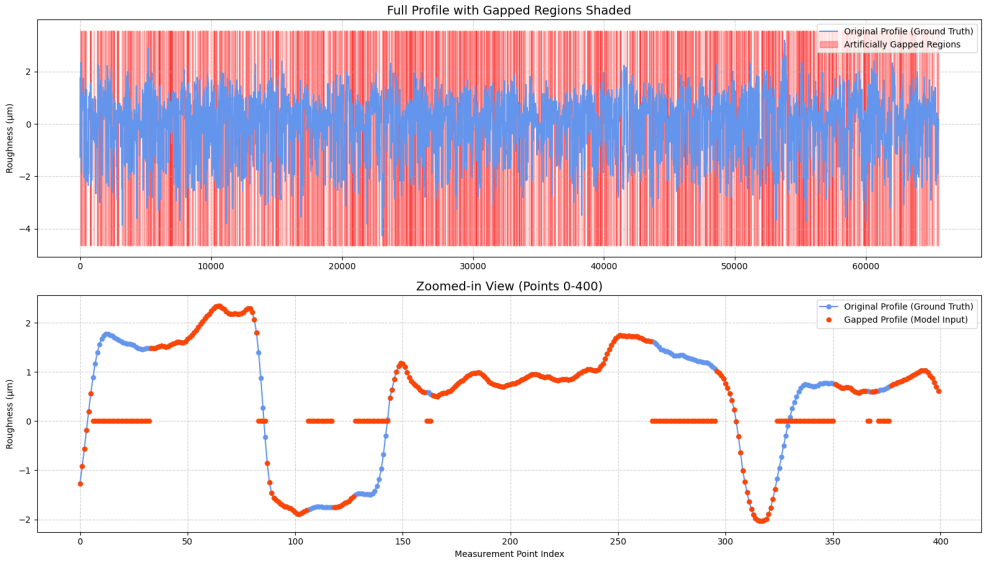


Figure 6.5: An example of a stylus profile with the gaps added. On top is the full profile with the original in blue and the gapped profile in red after the stochastic gap simulation process has been applied. Underneath is a zoomed-in view showing the first 0-400 measurement points. This illustrates the input data for the reconstruction models.

The analysis is performed on roughness profiles derived from high-fidelity stylus profilom-

etry measurements. For each raw surface profile, a waviness profile is calculated using a Fast Fourier Transform (FFT) based low-pass filter. The fundamental definitions of surface roughness parameters, including the formula for R_a and the ISO 4288 standard, are detailed in Chapter 2. In accordance with the ISO 4288 standard for surfaces in the expected R_a range of our samples (0.1 µm to 2.0 µm), a cutoff wavelength (λ_c) of 0.8 mm is used. The final roughness profile is obtained by subtracting this waviness profile from the raw data.

To simulate realistic data loss, artificial gaps are introduced into these clean roughness profiles. The properties of these gaps are designed to mirror those observed in the laser-scanner data. For each profile, a total percentage of missing data is drawn from a normal distribution with a mean of 28.6% and a standard deviation of 9.7%. This total is filled by creating multiple contiguous gap "chunks," whose individual lengths are sampled from an empirically derived distribution with a median length of 7 points and a mean of 12.6 points. To ensure robust and generalisable results, the dataset is divided into training, validation, and test sets using a grouped split based on the unique steel sample ID. To rigorously assess the performance and stability of the baseline interpolation methods, their experimental process was repeated 25 times with different random seeds for data splitting and gap generation. For the computationally later intensive deep learning models, a single representative data split and gap generation seed was used for the final hyperparameter comparison to ensure tractability.

6.3.2 Evaluation Framework

Performance is evaluated using several metrics designed to capture different aspects of reconstruction quality. To provide a comprehensive picture, these metrics are applied at two distinct scopes: **globally** (assessing performance on the full, reassembled profile) and **locally** (assessing performance only on the imputed points within the gaps).

- **Global Metrics (on the full profile):** These metrics assess the overall quality of the final, complete surface profile.
  - **Global R_a Error:** The Root Mean Squared Error (RMSE) between the R_a of the original, complete profiles and the R_a of the fully reconstructed profiles. This is the primary metric for assessing the preservation of the overall surface statistic.
- **Local Metrics (within gaps only):** These metrics assess the specific performance of the imputation algorithm itself.

- **In-Gap R_a Error:** The RMSE between the true R_a of the points within the original gaps and the In-Gap R_a of the imputed points (using the proxy calculation defined in section 6.3.2.1). This measures how well the local texture is statistically reconstructed.
- **In-Gap Pointwise Error (MSE/MAE):** The MSE and MAE calculated using only the reconstructed points and their original values within the gaps. This quantifies the positional accuracy of the imputation itself.

6.3.2.1 A Proxy for In-Gap Roughness (R_a)

When evaluating an imputation, pointwise metrics like MSE are calculated directly on the imputed values, a straightforward process that excludes the unchanged points in the series. However, assessing the statistical properties of *only* the imputed segment is more complex. Calculating the R_a of the entire reconstructed profile is one option, but the resulting error metric is often diluted, as the large number of unchanged points can mask significant errors within the smaller imputed regions. Conversely, calculating a standard R_a on just the points within a gap is problematic, as the local mean of a small, imputed segment can be highly unstable and may not reflect the true mean of the surrounding profile, leading to a skewed roughness value. A more stable approach is therefore needed to fairly evaluate the texture of the imputed points.

To address this, we define a proxy metric, hereafter referred to as the In-Gap R_a. This proxy calculates the average roughness of the imputed points, but critically, it measures their deviation from the mean of the entire segment (including the known context points). This provides a stable reference point and a more reliable measure of the imputed texture's consistency with the overall profile. The formula for the In-Gap R_a is given by:

$$R_{a,\text{In-Gap}} = \frac{1}{N_{gap}} \sum_{i=1}^{L} |z_s(i) - \bar{z}_s| \cdot M_s(i) \tag{6.2}$$

where z_s is a full sequence of length L, M_s is a boolean mask where $M_s(i) = 1$ for points within a gap, N_{gap} is the number of points in the gap, and $\bar{z}_s$ is the mean of the entire sequence z_s (both observed and missing points).

To validate this proxy as an accurate measure for evaluation, we devise a formal analysis on the ground-truth data prior to any imputation. The methodology for this test involves comparing the statistical distributions of two distinct values calculated from the same data segments. The first value is the standard R_a of an entire, complete data segment. The second is the In-Gap

R_a proxy, which is calculated using only the points that have been designated to form a gap within that same segment.

Our central hypothesis is that these two distributions should be nearly identical. This is based on the fact that the gaps are generated using the stochastic process detailed in Section 6.3.1, where the gap properties are sampled from distributions derived from the real-world laser data. While the R_a of any single, small gap might deviate from its parent segment due to local variations, it is expected that when aggregated across the entire large-scale dataset, the statistical properties of the gapped subsets will converge with those of the full segments. Therefore, the two calculated distributions should be statistically indistinguishable.

The analysis of these distributions across the training, validation, and test sets, as summarised in Table 6.2, confirms this hypothesis. The distributions are revealed to be statistically indistinguishable. For example, in the training set, the mean "Segment R_a" is 1.1920, while the mean "In-Gap R_a" is 1.1918. Similar strong agreement was observed across all quartiles and for the full profile analysis. This strong agreement confirms that the In-Gap R_a is a valid and robust metric for evaluating the statistical accuracy of the imputed regions for all methods, including the baselines.

Table 6.2: Comparison of the statistical distribution of "Segment R_a" vs. the "In-Gap R_a " on the segmented training data, demonstrating their near-identical properties.

Statistic	**Segment R_a**	**In-Gap R_a**
Mean	1.1920	1.1918
Std Dev	0.2994	0.3055
25th Percentile	0.9856	0.9798
Median (50th)	1.1771	1.1759
75th Percentile	1.3809	1.3847

6.4 Data Reconstruction Models and Techniques

6.4.1 Baseline Interpolation Methods

To develop and test robust imputation methods, it is first necessary to establish a rigorous benchmark using classical, well-understood techniques. Two classical interpolation methods were evaluated to serve as a performance benchmark:

- **Linear Interpolation:** Fills gaps by connecting the boundary points with a straight line.

- **Cubic Spline Interpolation:** Fits a piece-wise cubic polynomial to the known points to create a smoother, continuous reconstruction across the gap. For cases where gaps occur at the very beginning or end of a signal, linear interpolation is used as a fallback to avoid the explosive and incorrect predictions that can occur when extrapolating with high-order polynomials.

6.4.2 Machine Learning Based Reconstruction

Due to the computational and memory constraints of processing the full-length roughness profiles (65,535 points) directly with deep learning models, the data is first prepared for training through segmentation. Each full roughness profile is segmented into 64 non-overlapping segments of length 1,023. These shorter segments, with gaps introduced as previously described, form the individual input items for the models during training. However, the imputed segments are rebuilt into a whole imputed sample for evaluation and comparison with the baselines.

Data Pipeline and Augmentation The machine learning models are given data by a custom data pipeline designed to act as a form of on-the-fly data augmentation. For each item requested by the dataloader during training, the following operations are performed: 1) A full-length, clean stylus profile is selected; 2) It is segmented into 64 non-overlapping segments of length 1,023; 3) The stochastic gap simulation process, as detailed in Section 6.3.1, is applied to each individual segment; and 4) A single segment is then selected for that training step, while the remaining 63 are discarded. The specific segment is chosen deterministically based on the current epoch number; for an epoch e, the segment index i is calculated as $i = e \pmod{64}$. In the context of this study, a single training 'epoch' is therefore defined as a pass in which the model is shown one such segment from each full-length profile in the training set. Consequently, a full pass through every unique segment of the dataset requires 64 epochs. Crucially, due to the on-the-fly gap generation, each time this 64-epoch cycle repeats, every segment is presented to the model with a new, unique pattern of missing data, providing a continuous source of data augmentation throughout training.

Training Strategy and Validation Frequency All models are trained using a cosine annealing learning rate schedule. The SAITS and BRITS models are trained for a fixed budget of 200 epochs. The training budget for the generative CSDI model was extended, as its objective of learning the underlying data distribution benefits significantly from exposure to the widest

possible variety of augmented samples. The on-the-fly data loader ensures that each full pass through segments of the dataset (i.e., every 64-epoch cycle) presents a completely new set of gap patterns, and a longer training budget allows the model to learn from many such unique sets. This extended training showed little to no benefit for the other, non-generative architectures. A key distinction exists between training and validation data handling. During training, one segment per profile is yielded sequentially per epoch, each with a new stochastic gap simulation. For validation, however, a stable and comprehensive performance metric is required. Therefore, during a validation step, all 64 segments from every profile in the validation set are processed. Furthermore, the gap simulation for this validation set is made deterministic by using a fixed random seed. As a result of processing all segments, a single validation epoch is significantly more computationally expensive than a training epoch. Given this computational cost, the training loop for all models was optimised to perform validation only every 10 epochs to ensure the hyperparameter sweeps remained practical.

To provide a comprehensive evaluation of modern imputation techniques, a suite of state-of-the-art models was selected for comparison. The chosen models represent distinct architectural families—recurrent, attention-based, and generative—allowing for a robust comparison of different approaches to modelling the temporal dependencies and underlying data distributions required for roughness profile reconstruction.

BRITS (RNN-Based Model) BRITS (Bidirectional Recurrent Imputation for Time Series) is selected as a representative of state-of-the-art RNN-based imputation methods. The model treats the imputation task as a regression problem, where the values at missing timesteps are correlated with observed values. It utilises a Bidirectional Recurrent Neural Network, meaning it processes the time series in both forward and backward directions to capture a richer temporal context. A key innovation of BRITS is its use of a temporal decay mechanism, where the influence of past observations fades over time. This allows it to explicitly model the time gap between observations and better handle long, contiguous missing segments.

SAITS (Attention-Based Model) SAITS (Self-Attention based Imputation for Time Series) represents the state-of-the-art in attention-based imputation. In contrast to the sequential processing of RNNs, SAITS leverages a self-attention mechanism, allowing it to weigh the importance of all other observed data points in the sequence simultaneously when imputing a missing value. This enables the model to capture complex, long-range dependencies that can be challenging for recurrent models. SAITS is trained with a joint optimisation objective that

combines a direct imputation loss with a masked prediction task, forcing the model to learn rich temporal representations of the data.

CSDI (Diffusion-Based Model) We also evaluate CSDI (Conditional Score-based Diffusion Models for Imputation), which is a generative diffusion model. CSDI frames imputation as a conditional generation problem. It learns to reverse a diffusion process that gradually adds noise to the data. By conditioning this reverse process on the observed data points, it can generate realistic values for the missing segments that are consistent with the overall data distribution.

Additional Architectures Explored In the pursuit of a comprehensive architectural comparison, a Variational Autoencoder model with a Gaussian Process prior (GP-VAE) was also implemented and evaluated. This approach was explored for its potential to learn a robust, low-dimensional representation of the surface profiles and to provide a probabilistic, uncertainty-aware reconstruction. However, during extensive experiments and hyperparameter tuning, the GP-VAE would not train well and therefore consistently underperformed relative to the BRITS, SAITS, and CSDI architectures on both pointwise and statistical metrics. To maintain focus on the most competitive and promising techniques, further extensive experiments with this architecture were not pursued for the final analysis presented in this chapter.

Full Profile Reconstruction for Evaluation To calculate the final performance metrics for the machine learning models, the model's predictions on the individual segments must be re-assembled. For each full-length test profile, it is first broken down into its 64 constituent segments. The trained imputation model is applied to each segment individually. The reconstructed segments are then stitched back together in their original order to form a complete, full-length roughness profile. The final evaluation metrics are then calculated by comparing this fully reassembled profile to the original, complete ground-truth profile, ensuring a fair and accurate assessment.

6.4.2.1 Hyperparameter Optimisation

To ensure a robust and fair comparison, an extensive hyperparameter search was conducted for each machine learning architecture. Key parameters such as model depth (number of layers), embedding dimensions, and learning rate were varied. For the attention-based SAITS model,

the number of attention heads was also tuned. Table 6.3 summarises the search space for the main architectures.

Table 6.3: Summary of the hyperparameter search space for the evaluated machine learning models.

Model	Hyperparameter	Values Explored
BRITS	RNN Hidden Size	{256, 768, 1536}
	Learning Rate	{0.005, 0.001, 0.0005}
	Batch Size	{8, 16, 32}
SAITS	Model Dimension (d_{model})	{256, 512, 768}
	Number of Heads	{4, 8, 12}
	Number of Layers	{2, 4, 6}
	Learning Rate	{0.001, 0.0005, 0.0001}
CSDI	Number of Layers	{4, 6}
	Number of Heads	{4, 8}
	Learning Rate	{0.001, 0.0005, 0.0001}

6.4.3 Methodology for Selecting Optimal Models

The evaluation of the numerous hyperparameter configurations highlights a key challenge. The success of an imputation method in this industrial context is judged on two distinct criteria. The first is **pointwise accuracy** (measured by Pointwise RMSE): the imputed values should be as close as possible to the true, unobserved surface heights. The second, which is of particular importance to our industrial partner, is **statistical texture preservation** (measured by In-Gap R_a RMSE): the imputed segment must have a texture that is statistically consistent with the surrounding profile. As the following results will show, these two objectives are often in conflict, creating a multi-objective problem that requires a formal selection methodology.

To address this multi-objective problem rigorously, we employ a standard approach from the field of Multi-Criteria Decision Making (MCDM) known as the Distance from the Ideal Point method [56]. This technique is designed to identify the single candidate that represents the most effective compromise between competing objectives.

The method begins by defining a hypothetical "ideal point", p_{ideal}, in the 2D error space, which corresponds to the best possible score for each metric observed across all evaluated models. Since lower values are better for both error metrics, the ideal point is located at $(\min(RMSE_{pointwise}), \min(RMSE_{R_a}))$. A direct distance calculation to this point would be bi-

ased because the two metrics can operate on different numerical scales. To ensure each objective contributes equally, we first normalise the performance of all models to a common, dimensionless range of $[0,1]$ using Min-Max scaling. For an error metric x, the normalised value x_i' for the i-th model is calculated as:

$$x_i' = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{6.3}$$

In this new, normalised space, the ideal point is at the origin $(0,0)$. The Euclidean distance (L_2-norm) for each model to the origin is then calculated. For a given model with normalised performance (x_i', y_i'), the distance D_i is:

$$D_i = \sqrt{(x_i')^2 + (y_i')^2} \tag{6.4}$$

The model with the minimum distance D_i is selected as the best-balanced champion. This method provides a robust and unbiased criterion for identifying the single set of hyperparameters for each model configuration that offers the best overall compromise between the two key objectives, and is used to select the models presented in the following results section.

6.5 Results and Discussion

6.5.1 Baseline Methods Performance

The comprehensive analysis over 25 independent runs reveals a critical trade-off between the two baseline methods. The aggregate performance statistics are summarised in Table 6.4. When evaluating the global R_a error on the full reconstructed profile, Cubic Spline Interpolation significantly outperforms Linear Interpolation. Its mean percentage error is over four times smaller ($0.80\% \pm 0.27\%$ vs. $3.56\% \pm 0.17\%$).

Conversely, when examining the pointwise reconstruction error *within the gaps*, a more nuanced picture emerges. Linear Interpolation achieves a lower Mean Squared Error (MSE), while Cubic Spline Interpolation achieves a lower Mean Absolute Error (MAE). The higher MSE for spline interpolation suggests it is prone to producing larger individual errors (outliers), likely due to oscillations when fitting polynomials, even though its average error (MAE) is lower. This presents the core trade-off: spline interpolation is superior for preserving the global statistical properties of the profile (R_a), while linear interpolation is more conservative in avoiding large local pointwise errors. This sets a clear, non-trivial benchmark for the machine learning models to surpass.

Table 6.4: Statistical summary of baseline reconstruction performance across 25 independent runs. Metrics are separated by scope: global (on the full profile) and local (within gaps only). Values are reported as mean ± standard deviation.

Metric	**Linear Interpolation**	**Cubic Spline Interpolation**
Global Performance (on full profile)		
RMSE of Global R_a (µm)	0.0468 ± 0.0039	0.0120 ± 0.0038
Mean % Global R_a Error	$3.56\% \pm 0.17\%$	$0.80\% \pm 0.27\%$
Local Performance (within gaps only)		
RMSE of In-Gap R_a (µm)	0.1575 ± 0.0112	0.0427 ± 0.0121
Mean % In-Gap R_a Error	$12.54\% \pm 0.43\%$	$2.93\% \pm 0.94\%$
Pointwise RMSE	0.7413 ± 0.0532	0.7592 ± 0.0598
Pointwise MAE	0.4170 ± 0.0324	0.3498 ± 0.0299

The superiority of Cubic Spline is also evident when analyzing the statistical error in the gaps. The mean percentage error for the In-Gap R_a is substantially lower for Cubic Spline (2.93%) compared to Linear Interpolation (12.54%), as shown in Table 6.4. This confirms that, on average, it reconstructs a more statistically accurate texture within the missing segments themselves.

This statistical superiority, however, presents an apparent paradox. As illustrated in Figure 6.6, the pointwise predictions of Cubic Spline can be qualitatively poor, often producing noticeable oscillations and overshoots that are not present in the more conservative Linear Interpolation. To investigate how Cubic Spline can be statistically superior while appearing visually worse, an exhaustive head-to-head analysis was performed on over 9 million individual overlapping windows of 3,000 timesteps. For this analysis, we selected the single experimental (out of the 25) run where Cubic Spline had its worst statistical performance, providing the most challenging test case. Within each window, the method with the lower local In-Gap R_a error was declared the winner.

This analysis revealed that even in its worst run, Cubic Spline was the superior method in **72.76%** of all instances. The underlying reason for this is found in the nature of the R_a metric itself, which is further illuminated by a deeper dive into the window statistics. The analysis reveals a key difference in the statistical character of the errors: Linear Interpolation exhibits a strong and systematic negative bias, underestimating the true roughness in **93.10%** of all windows analyzed. In contrast, Cubic Spline produces a much more balanced distribution of errors, underestimating 41.07% of the time versus overestimating 58.93%. Because R_a is an

average-based metric, the random, balanced errors of Cubic Spline tend to cancel each other out, resulting in a global average that remains closer to the true value, despite its higher local pointwise errors.

The cause of Cubic Spline's large pointwise errors and poor visual quality is also quantified by the analysis. The average standard deviation of the Cubic reconstruction (1.5981) is higher than that of the original signal (1.5727), which numerically confirms the presence of oscillations. Conversely, the Linear reconstruction's average standard deviation is lower (1.5222), confirming its smoothing effect. Furthermore, the maximum single-window overestimation error for Cubic (R_a of 0.4645) was over four times larger than the maximum for Linear (R_a of 0.1147), accounting for its propensity for large, visually jarring pointwise errors.

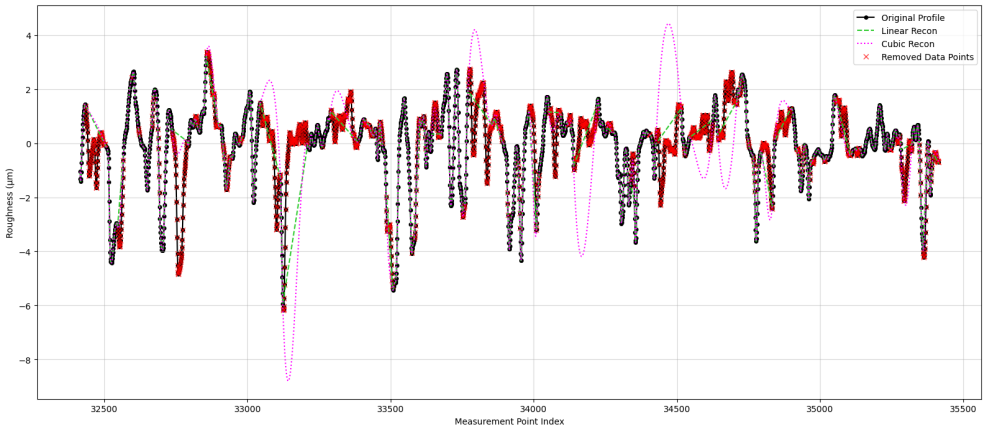


Figure 6.6: Visual comparison of reconstruction methods showing a zoomed-in section where Cubic Spline exhibited high pointwise MSE. Cubic spline interpolation (pink) can occasionally overshoot the true profile (black), whereas Linear interpolation (orange) tends to undershoot.

6.5.2 Machine Learning Methods Performance

To establish a performance benchmark, the machine learning models are first trained using a standard pointwise loss function (e.g., MSE or MAE) across a wide range of hyperparameters. While a detailed sensitivity analysis of all runs is beyond the scope of this chapter, the general findings indicated that larger models tended to perform better, albeit with diminishing returns, and that the learning rate was the most sensitive parameter for achieving stable convergence.

Crucially, the success of an imputation method in this industrial context is judged on two distinct criteria. The first is **pointwise accuracy**: the imputed values should be as close as possible to the true, unobserved surface heights, as measured by Pointwise RMSE. The second,

Table 6.5: Performance comparison of the best-balanced ML models trained with a standard MSE loss against baselines. All metrics are calculated on the imputed points within the gaps. Bold values indicate the best performance in each column and *italic* indicates second place.

Method	**Pointwise RMSE**	**In-Gap R_a RMSE (μm)**
Baselines		
Linear Interpolation	0.7413	0.1575
Cubic Spline Interpolation	0.7592	*0.0427*
Machine Learning Models		
BRITS (RNN)	*0.6856*	0.1966
SAITS (Attention)	**0.6419**	0.0643
CSDI (Diffusion)	0.7642	**0.0345**

which is of particular importance to our industrial partner, is **statistical preservation**: the imputed segment must have a texture that is statistically consistent with the surrounding profile, as measured by the In-Gap R_a RMSE. As the following results will show, these two objectives are often in conflict.

Following the extensive hyperparameter search, the single best-balanced model for each architecture was selected using the methodology detailed in Section 6.4.3. The performance of these champion models is compared against the baselines in Table 6.5. These results are also visualised in Figure 6.7, which plots statistical preservation against pointwise accuracy.

The results reveal a clear division of strengths among the different model families. As shown in Figure 6.7, models that excel at one metric do not necessarily lead in the other. In terms of pointwise accuracy, the attention-based **SAITS model is the definitive leader**, achieving the lowest in-gap Pointwise RMSE of **0.6419**. This significantly outperforms the RNN-based BRITS model (0.6856), which still beats the baselines for both interpolation baselines but not as much, demonstrating the effectiveness of self-attention for capturing the complex dependencies required for positional reconstruction. On the other hand the diffusion-based CSDI model performed worse in terms of pointwise accuracy than the other ML models and the baselines the only ml model of the three tested to do so.

This highlights a fascinating trade-off: while Cubic Spline is statistically accurate, its imputations often appear unrealistic due to oscillations (see Figure 6.6). CSDI, in contrast, produces imputations that are both statistically strong and qualitatively more realistic. This makes it a very strong contender for applications where both the statistics and the physical realism of the profile are important.

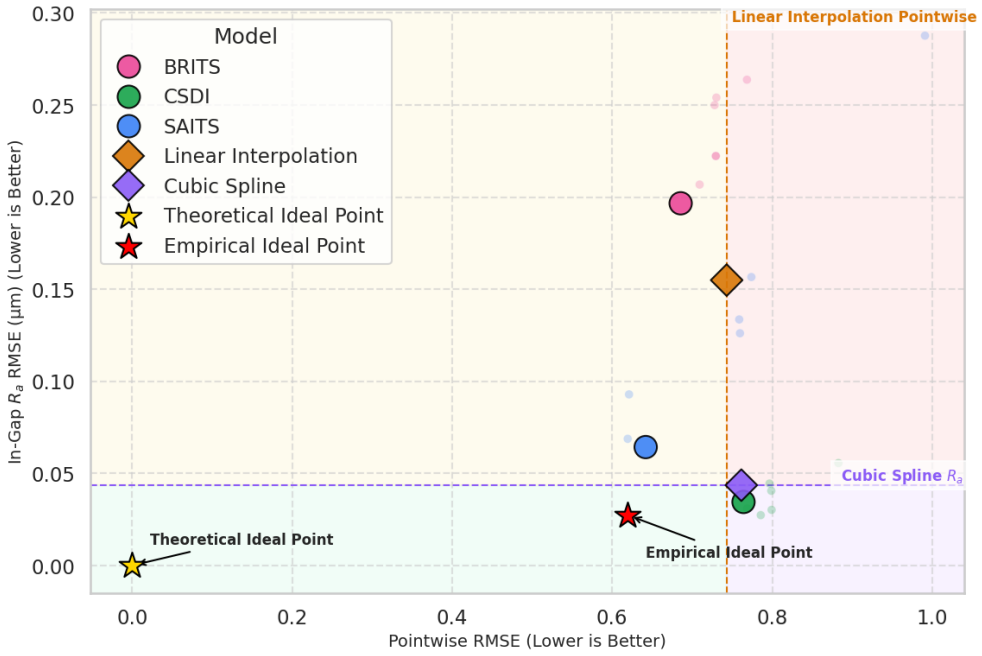


Figure 6.7: Performance trade-off between pointwise accuracy and statistical (R_a) preservation for models trained with a standard loss. Each small point represents a single hyperparameter configuration, while larger opaque points denote the best-balanced model for each architecture. This "best-balanced" selection is based on minimising the distance to the Empirical Ideal Point (a red star), which represents the best observed performance for each metric within the dataset. The plot also includes the Theoretical Ideal Point (a gold star at (0,0)), representing absolute perfection. The two baselines are shown as diamonds. The vertical, amber dashed line represents the superior pointwise RMSE achieved by the Linear Interpolation baseline, while the horizontal, purple dashed line indicates the superior R_a RMSE from the Cubic Spline baseline. These lines divide the plot into four colored quadrants: models in the amber quadrant outperform only the pointwise baseline, while those in the purple quadrant outperform only the R_a baseline. The green quadrant (bottom-left) highlights models superior to both, and the red quadrant (top-right) contains models dominated by the baselines.

However, when evaluating the preservation of the surface texture via the In-Gap R_a RMSE, the opposite picture emerges. Here, the generative **CSDI model is the strongest performer**, achieving an In-Gap R_a RMSE of only **0.0345 μm**. This surpasses the strong Cubic Spline baseline (0.0427 μm), a feat the other machine learning models fail to do. This quantitative result is further contextualised by a qualitative inspection of the imputations, as shown in Figure 6.8.

This visual analysis highlights a fascinating trade-off. While Cubic Spline performs well statistically, its reconstructions often appear unrealistic due to the characteristic oscillations of high-order polynomials. The CSDI model's imputations, in contrast, appear far more physically plausible. Although not perfectly accurate on a point-by-point basis, the texture it gen-

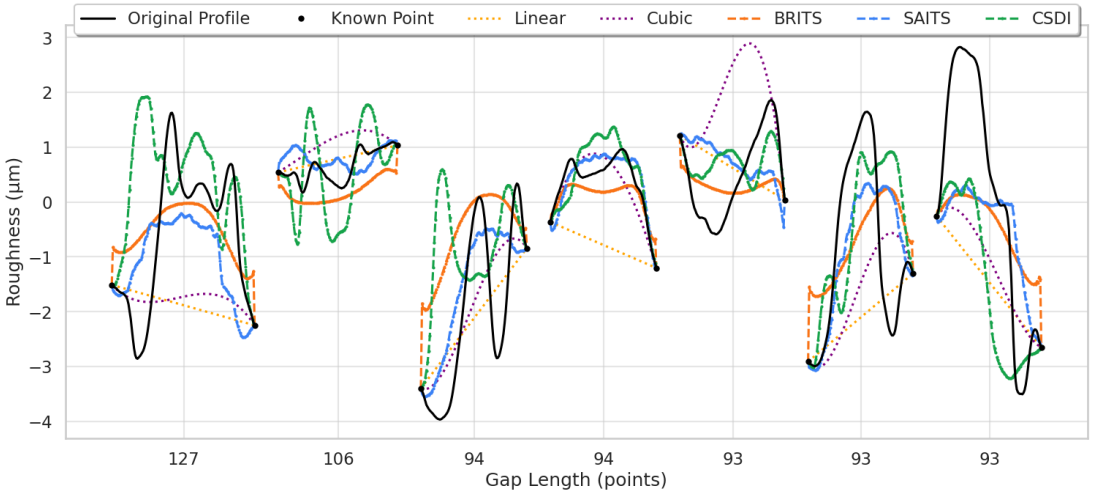


Figure 6.8: A qualitative comparison of imputation methods across the seven longest gaps identified in the single test sample on which the best-balanced SAITS model (the pointwise champion) exhibited its highest reconstruction error. The gaps are presented side-by-side, sorted by length in descending order, to provide a balanced view of model performance on substantial data losses. Each segment includes one known data point at its start and end to give context for the imputation.

erates is more consistent with the original signal. This is a crucial finding; as a generative model, CSDI's objective is to learn the underlying data distribution, which makes it inherently well-suited to recreating a realistic texture. While other models like SAITS also produce more realistic-looking imputations compared to the baselines, CSDI stands out as a strong overall contender by performing well both quantitatively on the statistical metric and qualitatively on visual realism. This makes it a compelling choice for applications where both the statistics and the physical plausibility of the profile are important.

This demonstrates a crucial challenge: models optimised for pointwise regression are not inherently the best at preserving statistical distributions, and generative models like CSDI can capture textural properties more effectively even if individual point predictions are less precise.

This trade-off between the pointwise-specialist SAITS and the statistical-specialist CSDI is the key finding from this initial analysis. The visualisation in Figure 6.7 powerfully illustrates this conclusion. The dashed lines, representing the best performance achieved by a baseline in each dimension, divide the space into performance quadrants. Notably, while SAITS and BRITS occupy the 'pointwise superior' quadrant and CSDI occupies the 'statistically superior' quadrant, no model resides in the bottom-left 'ideal' quadrant that would signify unequivocal superiority over both baselines. This vacancy reveals a fundamental limitation of SAITS and BRITS with a standard pointwise loss function for this task: while they successfully guide

these models to high positional accuracy, they fail to encourage the preservation of statistical texture. This strongly motivates the development of a loss function that is explicitly aware of the target statistics, which is explored in the subsequent section.

6.5.2.1 Improving Statistical Fidelity with a Statistics-Aware Loss Function

The preceding analysis revealed that models trained to produce a good positional accuracy with a standard pointwise loss did not perform well at statistical preservation in models trained. To overcome this limitation, we introduce a statistics-aware hybrid loss function, designed to create a hybrid objective that allows the model to be tuned for both goals simultaneously with a controllable training objective. The total loss is formulated as:

$$L_{\text{total}} = L_{\text{pointwise}} + \lambda_{R_a} \cdot L_{R_a,\text{In-Gap}}$$

where $L_{\text{pointwise}}$ is the standard pointwise error (e.g., MSE or MAE), $L_{R_a,\text{In-Gap}}$ is the squared error of the In-Gap R_a proxy, and λ_{R_a} is a weighting hyperparameter that controls the influence of the statistical penalty. Once again, we make use of the In-Gap R_a proxy defined earlier, as calculating R_a for the entire reconstructed profile severely obscures errors within the imputed regions, leading to a model that does not perform optimally at this task.

To evaluate the efficacy of this hybrid loss, an ablation study is conducted by training the SAITS and BRITS architectures with varying λ_{R_a} values. The results, where the best-balanced model for each configuration is selected, are summarised in Table 6.6 and visualised in Figure 6.9, demonstrating that incorporating this domain-specific R_a metric into the loss function successfully improves performance and enables precise control over the model's performance profile.

This performance trade-off is highlighted explicitly in Table 6.7, which separates the best-performing models for each objective, in contrast to the "best-balanced" models in Table 6.6. This clearly illustrates that the model which is best at pointwise accuracy (e.g., SAITS with $\lambda_{R_a} = 0.5$) is not the same model that is best at preserving statistics (e.g., SAITS with $\lambda_{R_a} = 1.0$).

For the attention-based SAITS model, the hybrid loss yields a remarkably favourable trade-off. As quantified in Table 6.8, setting $\lambda_{R_a} = 1.0$ achieves the best statistical fidelity, reducing the In-Gap R_a RMSE by a substantial **73.75%** compared to the pointwise only loss model ($\lambda_{R_a} = 0$). This significant gain comes at the negligible cost of a mere **1.44%** increase in pointwise error. The effectiveness of this trade-off is visually confirmed in Figure 6.9, where

Table 6.6: Ablation study on the effect of the R_a loss weight (λ_{R_a}). For each configuration, the single best-balanced model was selected using the minimum Euclidean distance method. Bold values indicate the best overall performance in each column across all methods. An asterisk (*) denotes the best result for that model's specific architecture. A dagger (†) marks the best-balanced model for SAITS and BRITS. N.B. The SAITS (†) model achieves the best overall performance.

Model	λ_{R_a}	Pointwise RMSE	In-Gap R_a RMSE (μm)
SAITS	0.0	0.6419	0.0643
	0.5	0.6497	0.0289
	1.0	0.6511	***0.0169**
	10.0†	***0.6269**	0.0321
BRITS	0.0	*0.6856	0.1966
	0.5	0.6932	0.1561
	1.0†	0.7002	0.1325
	10.0	0.8518	*0.0630
CSDI (Diffusion)		0.7642	0.0345
Cubic Spline Interp.		0.7592	0.0427
Linear Interp.		0.7413	0.1575

Table 6.7: Analysis of the performance trade-off within each model architecture. For each model family (SAITS, BRITS, and CSDI), this table identifies two champion models: the single best-performing experimental run when optimising solely for Pointwise RMSE, and the single best run solely for In-Gap R_a RMSE. This comparison highlights the performance trade-off inherent in the task: the model that is best at pointwise accuracy is not the same model that is best at preserving the R_a statistic. The corresponding score on the other metric is also shown. Bold values indicate the best overall performance across all architectures in each respective column.

Selection Criterion	Champion Model	Pointwise RMSE	In-Gap R_a RMSE (μm)
SAITS (Attention)			
Best Pointwise Only	SAITS ($\lambda_{R_a} = 0.5$)	**0.6140**	0.0488
Best R_a Only	SAITS ($\lambda_{R_a} = 1.0$)	0.6511	**0.0169**
BRITS (RNN)			
Best Pointwise Only	BRITS ($\lambda_{R_a} = 0.0$)	0.6856	0.1966
Best R_a Only	BRITS ($\lambda_{R_a} = 10.0$)	0.8518	0.0630
CSDI (Diffusion)			
Best Pointwise Only	CSDI	0.7642	0.0345
Best R_a Only	CSDI	0.7857	0.0273

the SAITS models with $\lambda_{R_a} > 0$ are pushed into the ideal bottom-left quadrant, simultaneously outperforming the strong Cubic Spline and CSDI baselines on both metrics. Furthermore, when using the ideal point as a measure of selecting the best model, the overall best-balanced model (marked with a †in Table 6.6) is found with $\lambda_{R_a} = 10.0$. This model achieves the best pointwise RMSE with a **-2.34% reduction** with a **-50.06% reduction** in In-Gap R_a.

Counter-intuitively, introducing the statistical loss term can also improve positional accuracy, suggesting it acts as a form of regularisation. This effect is most evident for SAITS and is highlighted by the red ellipse in Figure 6.9. The robust Top-3 analysis in Table 6.8 shows that with $\lambda_{R_a} = 0.5$, SAITS not only improves its R_a fidelity by **42.80%** but also slightly **improves its pointwise accuracy by 0.04%**. By penalising statistically unrealistic reconstructions, the R_a loss term appears to guide the model towards learning more generalisable features of the surface texture, preventing it from converging to an overly smooth solution that can be positionally inaccurate.

In contrast, the RNN-based BRITS architecture exhibits a much starker and less favourable trade-off. While the hybrid loss does improve its statistical fidelity, the gains come at a significant cost to pointwise accuracy. For instance, the best overall BRITS model $\lambda_{R_a} = 10.0$ had a **68%** gain in statistical performance is accompanied by a major **24.24%** degradation in pointwise RMSE, as shown in Table 6.8. This relationship is clear in Figure 6.9, where the BRITS models form a distinct pattern, moving sharply right (worse pointwise) as they move down (better R_a). This demonstrates that the loss function is successfully guiding the models to produce more statistically accurate R_a reconstructions but with a trade-off. Moreover, as seen in Table 6.6, even its best statistical performance (0.0630 µm) remains inferior to that of CSDI and the Cubic Spline baseline, highlighting the superior synergy between the attention mechanism and the hybrid loss objective and the model in general for this problem.

6.5.3 Conclusion

This chapter addressed the challenge of data gaps in on-line laser reflection measurements, which can compromise the accuracy of critical surface roughness statistics like R_a. First, we developed a formal, multi-metric methodology for identifying and characterising these poor-quality segments. This analysis, validated with our industrial partner, provided the empirical basis for the subsequent experiments, revealing that a significant portion of the data is compromised and establishing the statistical properties of the missing data.

To rigorously evaluate imputation techniques in a controlled environment, a framework us-

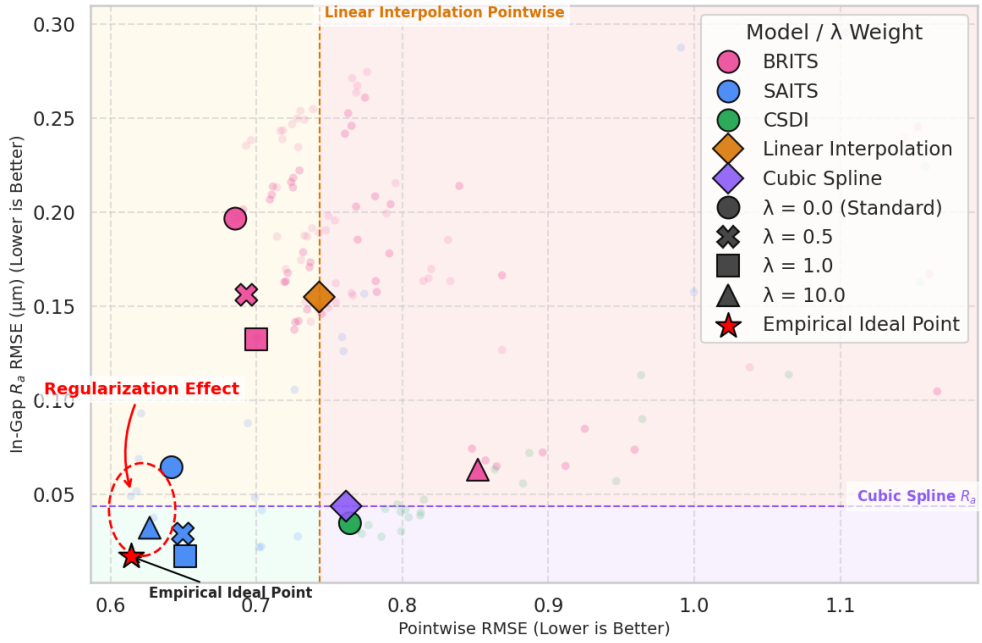


Figure 6.9: An ablation study of the effect of the R_a loss weight (λ) on the performance of BRITS, SAITS, and CSDI. Each small point represents a hyperparameter configuration, while larger points with different markers denote the best-balanced model for each λ value. The plot highlights a regularisation effect (red dashed ellipse), where SAITS models with an R_a loss term ($\lambda > 0$) achieved a better pointwise RMSE than the standard SAITS champion ($\lambda = 0$). CSDI is included for comparison using its default loss configuration. The interpolation baselines (diamonds) are shown by the dashed lines and colored quadrants, and the "Empirical Ideal Point" (red star) marks the best-observed performance across all runs.

ing gapped, high-fidelity stylus profilometry data was established. The initial analysis demonstrated that modern deep learning architectures, particularly the attention-based SAITS model, significantly outperform classical interpolation baselines in terms of pointwise reconstruction accuracy when trained with a standard loss function. However, this work also uncovered a crucial trade-off: these positionally accurate models failed to preserve the statistical texture of the surface, as measured by the In-Gap R_a RMSE, where the generative CSDI model and even the Cubic Spline baseline were superior.

To address this fundamental conflict between pointwise and statistical objectives, we introduced and validated a hybrid, statistics-aware loss function. By incorporating the In-Gap R_a metric directly into the training objective, we demonstrated that it is possible to guide the models to produce imputations that are not only positionally accurate but also statistically faithful. The ablation studies confirmed the efficacy of this approach, with the SAITS model trained with a combined loss ($\lambda_{R_a} = 10.0$) emerging as the overall best-balanced model, achieving a

Table 6.8: Quantitative trade-off analysis showing the percentage impact of the statistics-aware loss. A negative value indicates an improvement (error reduction), while a positive value indicates a degradation (error increase).

Analysis	Model	Pointwise RMSE Impact (%)	In-Gap R_a RMSE Impact (%)
Top-1	SAITS ($\lambda = 0.5$)	+1.21	-54.96
	SAITS ($\lambda = 1.0$)	+1.44	-73.75
	SAITS ($\lambda = 10.0$)	-2.34	-50.06
	BRITS ($\lambda = 0.5$)	+1.10	-20.61
	BRITS ($\lambda = 1.0$)	+2.13	-32.61
	BRITS ($\lambda = 10.0$)	+24.24	-67.96
Top-3 (Avg)	SAITS ($\lambda = 0.5$)	-0.04	-42.80
	SAITS ($\lambda = 1.0$)	+9.02	-61.65
	SAITS ($\lambda = 10.0$)	+4.11	-59.51
	BRITS ($\lambda = 0.5$)	+1.05	-20.37
	BRITS ($\lambda = 1.0$)	+2.04	-32.50
	BRITS ($\lambda = 10.0$)	+23.84	-66.11

state-of-the-art result that is strong on both metrics and capable of outperforming all baselines simultaneously.

These findings provide a strong foundation and a validated methodology for future work on the more challenging on-line laser reflection data. The principles learned here—particularly the effectiveness of attention mechanisms and the significant impact of a statistics-aware objective function—inform a clear path forward. A promising direction for future research would be to apply the trained models in a zero-shot transfer setting to the profiles generated by the closed-form solution, to assess if this improves their correlation with true stylus measurements. Furthermore, while this work relied on a robust function for identifying data gaps, another avenue for future work could be to frame the gap identification itself as a machine learning task, potentially leading to an even more powerful, end-to-end reconstruction pipeline.

Chapter 7

End-to-End Fine-Tuning of Pruned ROCKET Kernels for Efficient and Accurate Time Series Classification

The ROCKET algorithm represents a state-of-the-art approach for time series analysis, yet its reliance on a large ensemble of random, non-trainable kernels presents challenges for real-time applications and limits its ultimate performance. This chapter introduces a novel, two-part methodology to address these limitations. Part 1 details an engineering contribution: the development of an efficient, GPU-accelerated, and end-to-end differentiable ROCKET framework. Part 2 builds upon this framework to make a scientific contribution: it introduces and validates a prune-and-fine-tune pipeline that creates smaller, faster, and more accurate models by training a specialised subset of kernels. This work provides a complete methodology for transforming ROCKET from a static, random ensemble into a highly efficient and adaptable feature extractor.

7.1 Introduction

The ROCKET (RandOm Convolutional Kernel Transform) model has established itself as a top-performing algorithm for Time Series Classification (TSC), primarily due to its exceptional speed in training a linear classifier on features extracted from a large ensemble of fixed, randomly initialised convolutional kernels [28]. Its core innovation is the use of a large ensemble of 10,000 of these kernels to transform time series into a rich feature space, upon which

a simple and fast linear classifier can be trained. However, this architecture presents a fundamental trade-off. While the non-trainable nature of the kernels facilitates rapid training of the final linear layer, it introduces two significant challenges for real-world application and further performance enhancement.

Firstly, the computational cost of inference can be substantial. In deployment scenarios, every new data sample must be processed by the full ensemble of 10,000 kernels, a process that can be a bottleneck for applications requiring real-time predictions such as the on-line surface monitoring in steel manufacturing detailed in this thesis. Secondly, the kernels, being random and fixed, are not specialised to the target dataset, and the large number of kernels is likely to contain significant redundancy. This suggests that a smaller, more discriminative subset of kernels could potentially be more efficient without sacrificing, and perhaps even improving, predictive performance. Furthermore, the original ROCKET implementation is not end-to-end differentiable, precluding the possibility of fine-tuning the kernel weights themselves using gradient-based optimisation.

This chapter addresses these challenges through a two-part investigation that aims to evolve ROCKET from a static ensemble into a dynamic and specialised model. **Part 1** details an engineering contribution: the development of an efficient, GPU-accelerated ROCKET framework. This new implementation not only addresses the inference speed bottleneck but also introduces a novel differentiable pooling function, 'softPPV', a critical prerequisite for enabling end-to-end training. **Part 2** builds upon this framework by investigating a methodology for first applying a state-of-the-art pruning algorithm to reduce the large set of random kernels down to a computationally manageable subset, and then fine-tuning the weights of this pruned subset to create a model that is both highly efficient and more accurate than the original.

7.2 Part 1: An Efficient GPU-Accelerated Framework for ROCKET

7.2.1 Motivation: The Need for High-Speed Inference in Manufacturing

As established in the introduction to this thesis (Chapter 1), modern steel strip manufacturing requires real-time surface monitoring to enable fast feedback and in-process control to the temper mill. While the on-line laser measurement system provides the necessary data acquisition speed, the system used by our industrial partner is not sufficiently accurate for this task. Hence, the aim of this thesis is to apply machine learning models to fix and improve this transforma-

tion. The ROCKET model proves to be a highly accurate method for this task, as we later demonstrate in Chapter 8. Though preliminary results showed ROCKET's accuracy is sufficient, we also need to ensure that the model meets the engineering challenges of deploying it in a demanding, high-speed industrial environment. This section of the chapter focuses on the engineering challenges of deploying such a model in a demanding industrial environment. Having made adjustments to improve the model efficiency, we will then perform benchmark experiments with the UCR Archive datasets [27] to evaluate the speed of our GPU implementation against the simpler CPU-based version. We have chosen these datasets as they are the primary benchmark on which the ROCKET model was originally evaluated.

For any model to be viable in a fast feedback control loop, predictive accuracy is not the only consideration; the inference speed of the model is of paramount importance. Our in-production data collected from the steel surface is of high resolution, with each sample containing 20 channels and 50,000 timesteps. The production line speed can reach up to 300cm/s, which, given the sensor's sampling rate, means that up to 60 predictions per second are required to achieve real-time monitoring. This combination of large input data and high-frequency prediction demand creates a significant computational challenge.

The ROCKET model, introduced in Section 2.5.1, is a strong candidate for this task due to its proven performance on both Time Series Extrinsic Regression (TSER) problems [113] (most relevant to our problem setting), and Time Series Classification (TSC) problems, as demonstrated by its original evaluation on the UCR Archive datasets. However, while its training is fast, its inference is not. The original CPU-based implementation is unable to meet the 10-60 predictions/second requirement on our data identified above. This motivates the primary engineering goal of this section: to develop a GPU-accelerated implementation of the ROCKET transform that can meet stringent speed requirements of an industrial production line, and in doing so, create a more powerful and versatile tool for the wider research community.

7.2.2 Methodology: GPU Implementation and Differentiable Pooling

The methodology involves a complete reimplementation of the ROCKET feature transformation pipeline in PyTorch [95] to leverage GPU parallelisation. This framework was designed to be a faithful reproduction of the original, while introducing a key innovation to enable future work in kernel training.

7.2.2.1 ROCKET Architecture and Random Kernel Initialisation

We mimic the details of the original ROCKET specification, which was detailed in the background Section 2.5.1. Our implementation generates the default 10,000 1D convolutional kernels, adhering to the same random parameterisation for kernel length, weights, bias, dilation, and padding used in the original work. As a reminder, these parameters are generated as follows:

- **Length:** The kernel's length is chosen with equal probability from the set $\{7, 9, 11\}$
- **Weights:** The weights are sampled from a standard normal distribution, $\mathcal{N}(0,1)$, before being mean-centered (i.e. the mean of the weights is subtracted from each weight)
- **Bias:** The value for the bias term is sampled from a uniform distribution, $\mathcal{U}(-1,1)$
- **Dilation:** Dilation is sampled on an exponential scale $d = \lfloor 2^x \rfloor$ where $x \sim \mathcal{U}(0,A)$ and $A = log_2 \frac{l_{input}-1}{l_{kernel}-1}$, with l_{input} being the length of the input time series and l_{kernel} being the length of the kernel
- **Padding:** For each kernel, padding is randomly added with a probability of 0.5. If padding is used, zero-padding is added to the beginning and end of every time series when the kernel is applied

As with the original implementation, for multivariate time series, a random subset of input channels is convolved for each kernel. A key design feature is that our PyTorch implementation is fully compatible with kernels generated by the original CPU version, allowing for them to be directly loaded into the model for direct and fair comparisons. To ensure perfect reproducibility, kernels are generated once for each dataset for each run and then loaded by all models in the comparison in that run. For the steel dataset, a single set of 10,000 kernels is generated and used for all experiments.

7.2.2.2 Proportion of Positive Values (PPV) and Differentiable `softPPV`

A key feature of ROCKET is its use of the Proportion of Positive Values (PPV) pooling function. For a given vector z of length L (the output of a convolution), the PPV is the count of its positive elements divided by L. The original implementation does this using a for loop and an if statement that increments a counter if the element is larger than 0. We however express PPV

as the mean of the Heaviside step function $H(x)$ applied to each element of z:

$$H(x) := \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases} \tag{7.1}$$

$$\text{PPV}(z) = \frac{1}{L}\sum_{i=1}^{L} H(z_i) \tag{7.2}$$

The Heaviside step function is non-differentiable at $x = 0$, which prevents end-to-end training of the kernel weights using gradient-based methods. To overcome this, a novel differentiable version of PPV, termed "softPPV" is developed. This is achieved by approximating the Heaviside function with a sigmoid function:

$$H_{soft}(x;\lambda) = \frac{1}{1+e^{-\lambda x}} \tag{7.3}$$

However, to investigate potential training benefits of moving the saturation of the gradients in the flat part of the sigmoid curve, we developed a modified, shifted sigmoid, which we term $H_{shifted\_soft}(x;\lambda,\beta)$, as our primary activation:

$$H_{shifted\_soft}(x;\lambda,\beta) = \frac{1}{1+e^{-\lambda x-\beta}} \tag{7.4}$$

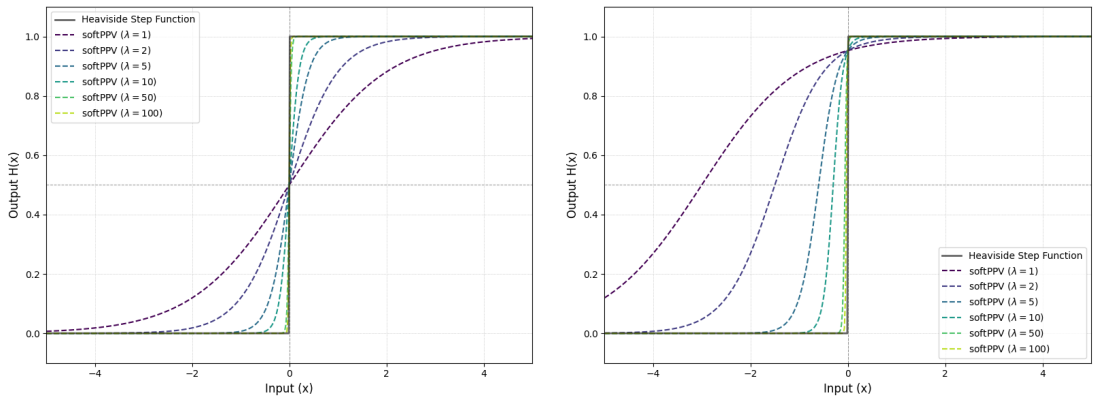
Throughout our experiments, we use a fixed shift of $\beta = 3$. This was chosen as it worked well in preliminary experiments. The final softPPV is the mean of these shifted sigmoid outputs:

$$\text{SoftPPV}(z;\lambda) = \frac{1}{L}\sum_{i=1}^{L} H_{shifted\_soft}(z_i;\lambda,3) \tag{7.5}$$

The hyperparameter λ controls the steepness of the sigmoid, which acts as a "softness" parameter. As $\lambda \to \infty$, the soft approximation approaches the true Heaviside function. We use a constant term of $\beta = 3$, which shifts the function's inflexion point to the left, which can improve gradient flow for distributions centred below zero. This differentiability is the critical property that enables the fine-tuning experiments in Part 2 of this chapter.

7.2.3 Results: Speed Benchmarks and Validation

A series of experiments is conducted to validate two key aspects of the GPU implementation: its computational speed advantage over the CPU-based implementation and the MiniROCKET model, and the predictive fidelity of the novel 'softPPV' function. The speed benchmarks are performed on both the UCR archive and industrial steel dataset with an analysis of the model's usability in the context of the industrial manufacturing problem.



A. Standard soft Heaviside ($\beta = 0$)

B. Shifted soft Heaviside ($\beta = 3$)

Figure 7.1: A visualisation of the Heaviside step function and the Soft varients for different values of the steepness parameter, λ. (a) shows the standard soft Heaviside, centred at $x = 0$. As λ increases, the sigmoid function becomes a closer approximation to the true non-differentiable Heaviside function (shown in black). (b) shows our shifted version used in experiments, with the inflexion point moved to the left.

7.2.3.1 Experimental Setup

For the steel dataset benchmarks, experiments are run on two primary testbed machines: one equipped with an AMD R5 3600x (6-core) CPU and an Nvidia 2080ti GPU, and a second, more powerful machine with an AMD R9 5950x (16-core) CPU. All experiments are run sequentially in a randomly permuted order. For the GPU-based PyTorch implementation, performance is tested across a range of batch sizes from 1 to 1000. This is because the performance benefit of processing multiple examples at once on the CPU tapers off much sooner due to its parallelisation limits, and extending the tests to larger batch sizes would be pointless and excessively time-consuming. It is worth noting that the parallelisation mechanism on the CPU implementation, which typically involves creating new threads, is distinct from the batching approach of a GPU using PyTorch. However, for ease of comparison, the term "batch size" is used consistently to refer to the number of time series examples transformed in a single function call.

For the UCR archive and synthetic data benchmarks, a different experimental setup is employed. These experiments are conducted on a single machine equipped with an AMD Ryzen Threadripper PRO 3975WX 32-Cores CPU and an Nvidia 3090 GPU. As these benchmarks are run with a different hardware configuration, their results are presented independently from the steel dataset experiments to avoid potentially misleading direct comparisons.

7.2.3.2 Computational Speed Benchmarks

The timing experiments, visualised in Figure 7.2, demonstrate a clear and significant performance benefit to performing the ROCKET transformation on a GPU. As the number of examples per batch increases, the PyTorch GPU implementation becomes substantially faster than the sktime CPU versions, a result of the superior parallelisation capabilities and faster floating-point computation of the GPU architecture.

Quantitatively, the slowest configuration is the sktime ROCKET implementation on the 6-core 3600x CPU, which reaches a peak speed of just 0.22 transformations per second (t/s). The more powerful 16-core 5950x CPU improves this peak speed to 0.65 t/s. In contrast, the GPU implementation using 'softPPV' achieves a peak speed of 5.12 t/s (at a batch size of 822), while the version using the standard Heaviside PPV reaches a peak speed of approximately 2.6 t/s. Comparing the fastest CPU result (0.65 t/s) with the fastest GPU result (5.12 t/s) reveals an approximately **688% speed increase**.

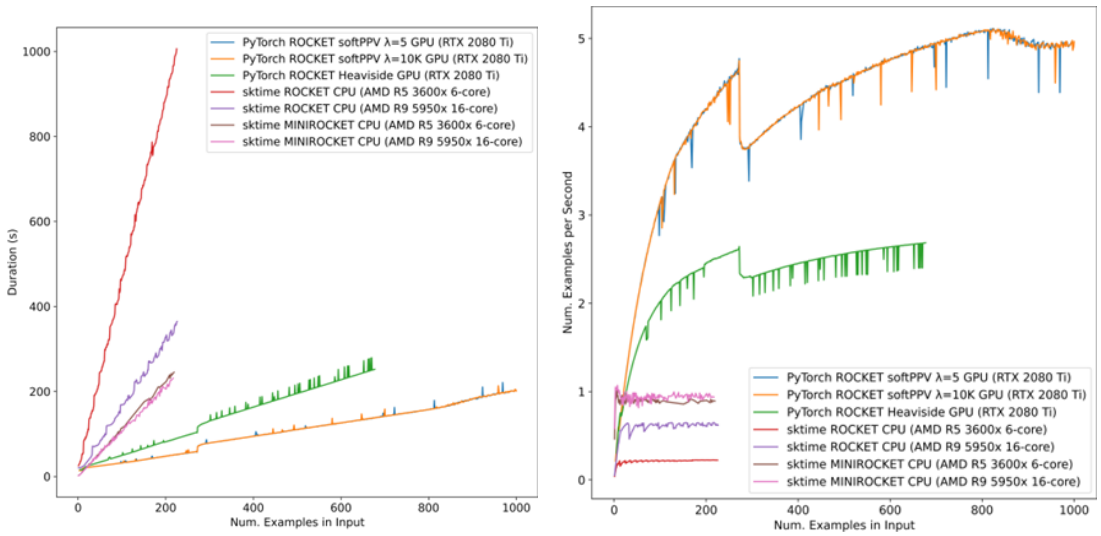


Figure 7.2: Timing the ROCKET transform duration for the PyTorch GPU implementation versus the 'sktime' CPU implementation. (a) Plots the total duration required to transform an input batch of increasing size. (b) Plots the effective speed in transformations per second (t/s), highlighting the performance plateaus of each configuration.

An analysis of the performance curves in Figure 7.2b reveals that all configurations reach a performance plateau, likely corresponding to the maximal capacity for parallelisation on the respective hardware. It is notable that even the significantly more powerful 16-core CPU plateaus at a much lower throughput than the GPU. The GPU results also exhibit two interest-

ing artefacts: a consistent, sharp drop in performance at a batch size of 273, and occasional large spikes in duration at very large batch sizes. The cause of the performance dip remains unknown, though the randomised experimental sequence rules out thermal throttling.

As this discontinuity occurs at the same batch size across all evaluated implementations, it is unlikely to be related to the specific PPV formulation. Each input sample consists of a $2^{16} \times 20$ tensor (approximately 5.2 MB in single precision), and increasing the batch size beyond this point substantially increases the overall memory footprint. Notably, the observed transition occurs close to a power-of-two boundary (close to 256 samples), which is commonly associated with changes in GPU block scheduling or occupancy behaviour. The behaviour is therefore likely due to a GPU resource threshold, such as a change in occupancy, register pressure, or memory allocation strategy triggered by the batch-dependent tensor size.

When compared to the highly optimised MiniROCKET algorithm, the GPU ROCKET implementation is initially slower for very small batch sizes. However, as shown in Figure 7.2a, the GPU implementation overtakes the CPU-based sktime MiniROCKET at a batch size of just 20. The MiniROCKET model peaks at 1.07 t/s on the fastest CPU, and interestingly, does not appear to scale as effectively with CPU core count as the ROCKET model does.

7.2.3.3 Predictive Fidelity of Differentiable Pooling

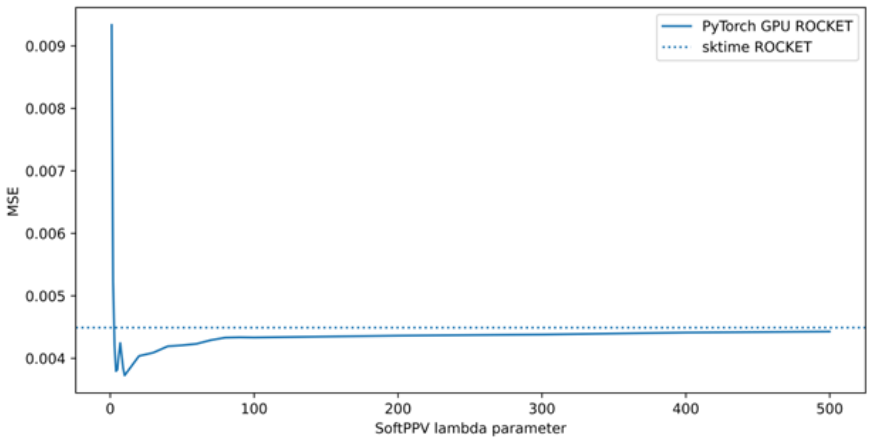


Figure 7.3: Comparison of the final test set MSE for a ridge regressor trained on features from different ROCKET transform implementations. The performance of the softPPV function is shown for a range of steepness hyperparameters (λ).

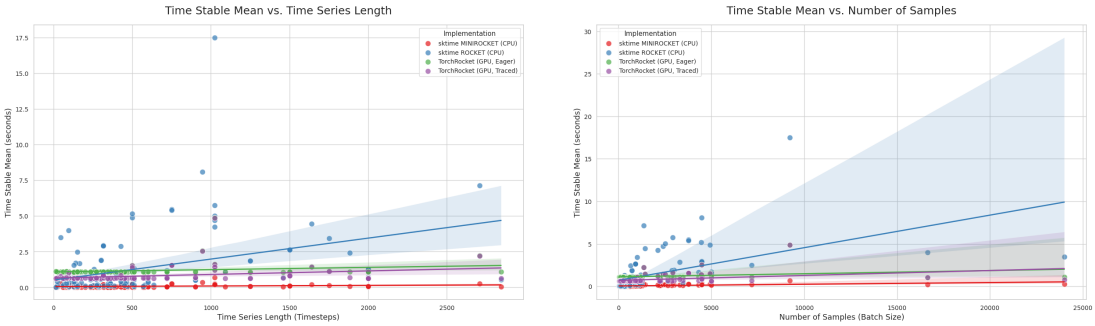
To validate that the differentiability introduced by the softPPV function does not come at the cost of predictive accuracy, a second experiment is conducted. A ridge regressor is trained

using the features generated by different ROCKET transform implementations: the original sktime version with standard PPV, and several versions of our PyTorch implementation, each with a different value for the softPPV steepness hyperparameter, λ. The Mean Squared Error (MSE) is then compared on a withheld test set.

The results, shown in Figure 7.3, confirm that using the softPPV approximation does not degrade performance. For values of $\lambda \geq 3$, the differentiable version produces slightly better MSE results than the original non-differentiable implementation. Specifically, softPPV with $\lambda = 1$ and $\lambda = 2$ produces a worse MSE (0.0093 and 0.0052, respectively), but $\lambda = 3$ (MSE 0.0042) and $\lambda = 4$ (MSE 0.0038) both outperform the sktime PPV baseline (MSE 0.0045). This is a crucial finding, as it demonstrates that differentiability can be achieved without sacrificing (and in this case, even marginally improving) the model's predictive power.

7.2.3.4 Validation on UCR Benchmark Datasets

To further validate the GPU framework and provide a broader benchmark for the academic community, timing experiments were also conducted on a selection of datasets from the UCR time series archive. As these experiments were run at a later date with different hardware and a refined timing methodology, the results are presented here independently to avoid exact direct but potentially misleading comparisons with the steel dataset benchmarks, although the same trends are seen.



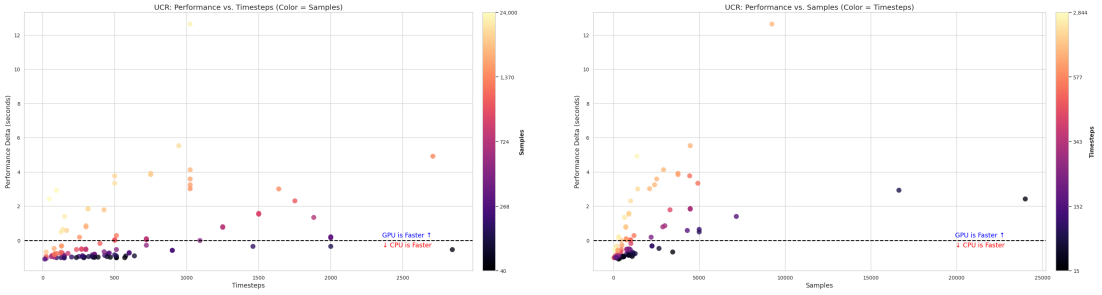
A. Mean inference time vs. time series length.

B. Mean inference time vs. number of samples.

Figure 7.4: Performance scaling on UCR datasets. Each point represents a dataset. While MINIROCKET is fastest, our GPU ROCKET (blue) quickly surpasses the standard CPU ROCKET (orange) as both time series length (a) and number of samples (b) increase. Note the significantly wider confidence interval for CPU ROCKET in (b), indicating higher performance variance as the number of samples grows.

A deep analysis of the UCR timing results reveals clear scaling characteristics for each

ROCKET implementation, as shown in Figure 7.4. When plotting inference time against time series length and number of samples, we observe that while CPU-based models are faster for very small inputs, our GPU ROCKET implementation quickly becomes more efficient than the standard CPU ROCKET as data size increases. The crossover point occurs at a relatively low number of timesteps. However, the CPU-based MINIROCKET remains the fastest across most of the UCR datasets, exhibiting a nearly flat performance curve that appears insensitive to time series length. The plot of time versus number of samples (Figure 7.4b) reveals a similar trend.



A. Performance delta vs. time series length.

B. Performance delta vs. number of samples.

Figure 7.5: Performance delta (CPU Time - GPU Time) for ROCKET on UCR datasets. Points above the zero-line indicate the GPU is faster. The GPU's advantage grows significantly with increasing time series length (a) and, even more noticeably, with number of samples (b), with very few CPU wins for batches larger than ~4000 samples.

This relationship is further clarified by analysing the performance delta (CPU Time - GPU Time), as shown in Figure 7.5. A positive delta indicates that our GPU implementation is faster. While the CPU models have an advantage on datasets with few timesteps or samples (the dense cluster of points below the line in both plots), a clear trend emerges. The advantage of the GPU implementation grows substantially with both increasing time series length (Figure 7.5a) and, even more noticeably, with an increasing number of samples (Figure 7.5b). This demonstrates the superior scaling of the GPU-based parallel processing.

Our benchmarks on the univariate UCR datasets revealed a complex performance picture. As shown in Figure 7.6, while our GPU ROCKET implementation shows a clear throughput advantage over the original CPU version as the number of samples increases, the highly-optimised MINIROCKET often remains faster on these specific single-channel datasets. This finding was initially puzzling, as it contradicted the significant speed-ups observed on our own industrial steel dataset. This discrepancy highlights that the UCR datasets differ from our specific use case not only in their univariate nature but also in their typically shorter time series lengths and smaller sample sizes. It is clear that to fully understand the dynamics and per-

formance trade-offs we need a more systematic investigation using synthetic data. To gain a complete understanding of how performance scales, we generate a large suite of synthetic datasets across a grid of parameters: number of samples, number of channels, and number of timesteps.

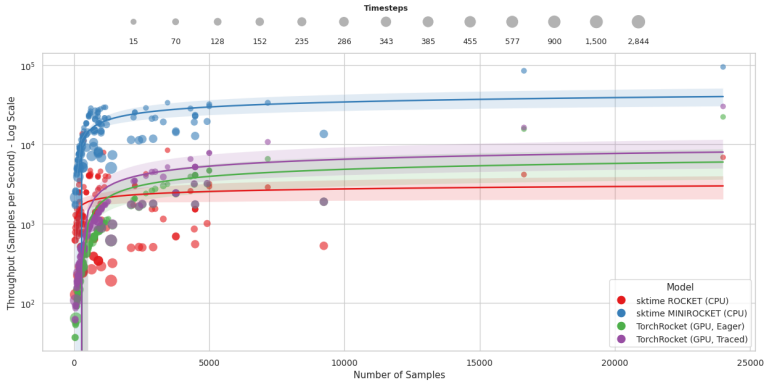


Figure 7.6: Throughput comparison of ROCKET implementations on single-channel UCR archive datasets. While GPU ROCKET outperforms CPU ROCKET, the CPU-based MINIROCKET often shows the highest throughput in this specific, univariate context.

7.2.3.5 Systematic Analysis with Synthetic Data

The benchmarks on the univariate UCR datasets and the difference to our dataset highlight the difficulty of characterising model performance on this connection alone, where dimensions such as time series length, number of samples, and number of channels are correlated and cannot be controlled independently. To build a complete picture of how each ROCKET im-

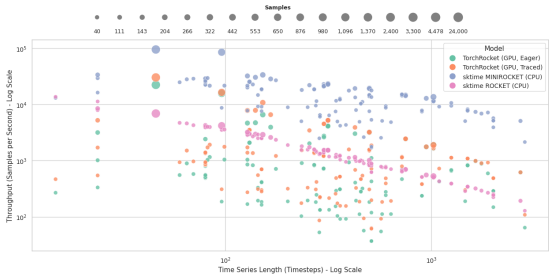


Figure 7.7: Throughput versus time series length for ROCKET implementations on univariate UCR datasets (log-log scale). A general negative trend is observed, where throughput decreases as time series length increases. MINIROCKET consistently achieves the highest throughput. The CPU ROCKET implementation has a very tight performance profile, whereas the GPU implementation shows a much wider variance, with performance within the spread being influenced by other dataset parameters like the number of samples.

plementation scales, we conduct a large-scale timing study on synthetic data. This approach allows us to systematically and independently evaluate the impact of each data dimension on performance by generating datasets across a comprehensive grid of parameters, as detailed in Table 7.1.

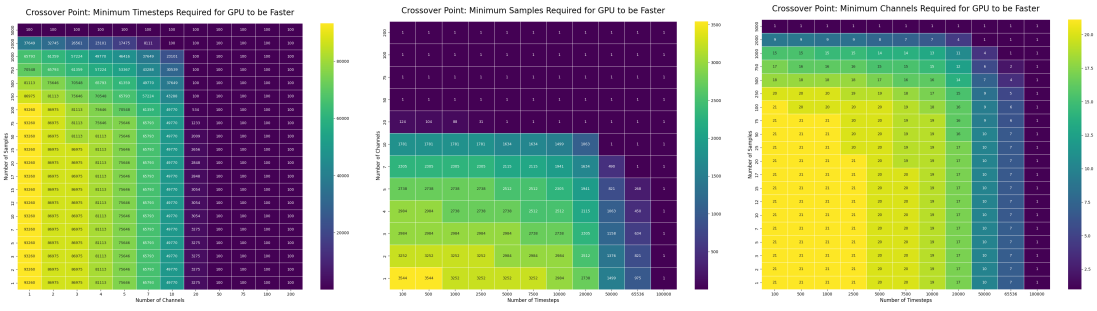
Table 7.1: Grid of parameters for synthetic data generation.

Parameter	**Values**
Number of Samples	1, 2, 3, 5, 7, 10, 12, 15, 17, 20, 25, 50, 75, 100, 250, 500, 750, 1000, 2000, 5000
Number of Channels	1, 2, 3, 4, 5, 7, 10, 20, 50, 75, 100, 200
Number of Timesteps	100, 500, 1000, 2500, 5000, 7500, 10000, 20000, 50000, 65536, 100000

To quantify the impact of these parameters on the runtime of each model, a regression analysis is performed on the timing results from the synthetic grid search. The analysis reveals the most impactful terms for each implementation. For the standard CPU ROCKET, the 'Samples * Timesteps' interaction term has the largest coefficient, confirming that its runtime scales significantly with the total amount of data processed. For our GPU ROCKET implementation, the coefficients are substantially smaller across the board, demonstrating its superior scaling, although the 'Samples * Timesteps' term remains the most significant factor. Crucially, the analysis for the CPU-based MINIROCKET reveals a different performance driver: the 'Channels * Timesteps' interaction is the most impactful term. This finding explains the discrepancy between the UCR and industrial dataset benchmarks; MINIROCKET's high efficiency on the univariate UCR datasets is not maintained as the number of channels increases, where its performance degrades more sharply than even the standard CPU ROCKET.

CPU vs. GPU Crossover Analysis To precisely map the performance boundaries between the CPU and GPU ROCKET implementations, we use the results from the synthetic data grid to fit a logistic regression. By solving for the decision boundary of this model (where the probability of the GPU being faster is 50%), we can determine the "crossover point" for any single parameter while holding the other two constant. The results of this analysis are visualised as heatmaps in Figure 7.8.

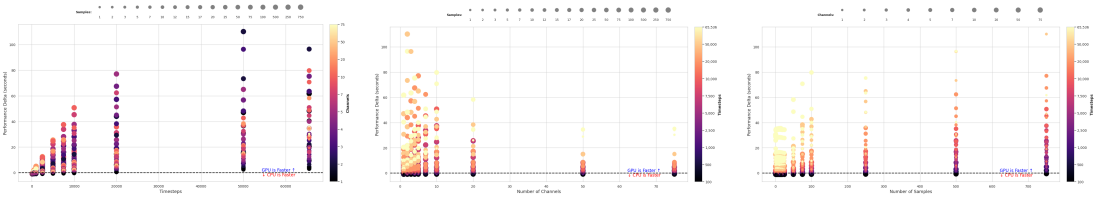
The analysis reveals that the crossover point is highly dependent on the dimensionality of the data, particularly the number of channels. Figure 7.8A. shows the minimum number of



A. Minimum timesteps. B. Minimum samples. C. Minimum channels.

Figure 7.8: Crossover point analysis showing the minimum value of a target parameter (timesteps, samples, or channels) required for our GPU ROCKET implementation to be faster than the standard CPU ROCKET. The heatmaps clearly show that the GPU's advantage grows significantly as data dimensionality increases.

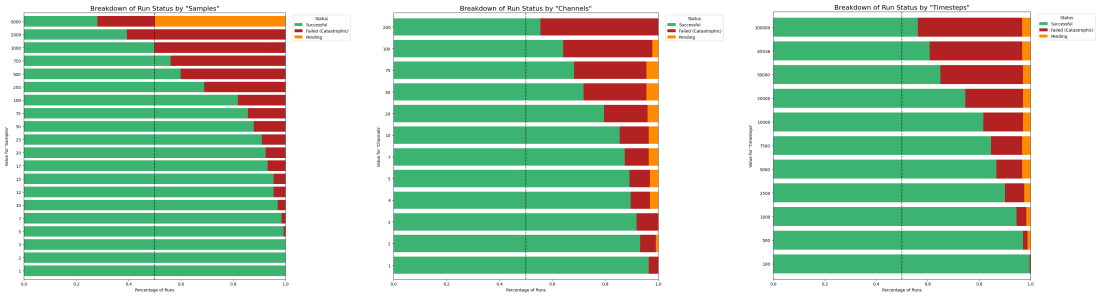
timesteps required for the GPU implementation to outperform the CPU version. For a small batch of 10 samples, the crossover point for a single-channel series is over 93,000 timesteps, explaining why the CPU version is competitive on many UCR datasets. However, for a 20-channel series (similar to our industrial data), this drops to just over 3,000 timesteps. For series with 50 or more channels, the GPU implementation is faster for almost any time series length. A similar trend is evident when analysing the crossover point for the number of samples (Figure 7.8B.) and channels (Figure 7.8C.), confirming that the GPU's parallel processing advantage grows rapidly with data volume and dimensionality. This systematically demonstrates why the GPU implementation is superior for the high-channel industrial data used in this thesis, despite the more complex performance picture on the smaller, univariate UCR benchmarks.



A. Performance vs. time series length. B. Performance vs. number of channels. C. Performance vs. number of samples.

Figure 7.9: Performance delta (CPU Time - GPU Time) for the ROCKET implementation across the synthetic data grid. Points above the zero-line indicate that the GPU is faster. Each subplot shows the delta against a primary parameter (x-axis), while using point hue and size to represent the other two parameters. The plots collectively demonstrate how the GPU's performance advantage grows as all three parameters—time series length, number of channels, and number of samples—increase.

To visualise these scaling characteristics directly, Figure 7.11 plots the total transform time against time series length and number of samples for a selection of the combinations. The anal-



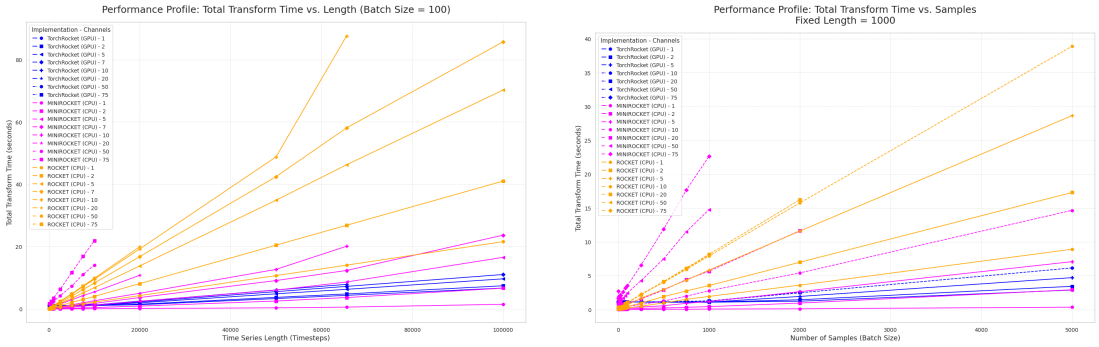
A. Failures by number of samples. B. Failures by number of channels. C. Failures by number of timesteps.

Figure 7.10: Breakdown of failed experimental runs (e.g., due to out-of-memory errors) across the synthetic data grid. The plots show the distribution of failures against the number of samples, channels, and timesteps.

ysis highlights the superior performance of the GPU implementation, particularly for large and complex datasets. As shown in Figure 7.11A., the runtime of the GPU ROCKET (blue lines) is nearly insensitive to the length of the time series, exhibiting an almost flat performance curve. In contrast, both CPU-based models show a significant increase in runtime with longer time series as channels gets larger, with the standard CPU ROCKET (orange lines) scaling much more poorly than MINIROCKET (magenta lines). A similar trend is evident when plotting performance against the number of samples (Figure 7.11B.). While MINIROCKET is highly competitive for very small batch sizes, the GPU implementation's scaling advantage becomes dominant as the number of samples increases for channels larger that 2. In both scenarios, increasing the number of channels increases the runtime for all models, but the effect is far less pronounced for the GPU implementation. These results confirm that for the large, multi-channel data typical of our industrial use case, the GPU implementation provides a substantial and necessary performance advantage.

7.2.3.6 Industrial Usability Analysis

Finally, the framework's performance is assessed in the context of its intended industrial application, focusing on the implications of its inference speed. The achieved peak throughput of approximately 5.5 t/s must be compared to the data generation rate of the production line, which requires between 10 (slowest line speed) and 60 (fastest line speed) predictions per second. While our implementation cannot provide a prediction for every single laser reading at the highest line speed, it is capable of providing predictions for a large percentage of them, and can comfortably produce 50% coverage with the slower line speeds. This provides



A. Scaling with time series length.

B. Scaling with number of samples.

Figure 7.11: Total transform time for ROCKET implementations as a function of time series length (a) and number of samples (b). The GPU ROCKET (blue) shows superior scaling compared to the CPU ROCKET (orange) and CPU MINIROCKET (magenta), especially for longer time series and larger batch sizes with larger channel counts.

line operators with the near-real-time feedback necessary for in-process control, a significant improvement over the existing state-of-the-art; whereas the closed-form solution provided by the sensor hardware was fast but too inaccurate to be usable, our GPU-accelerated approach enables a model with usable accuracy (as demonstrated in Chapter 8) to run at a speed that provides near-real-time feedback to line operators.

7.3 Part 2: Pruning and Fine-Tuning for Enhanced Accuracy and Efficiency

The framework developed in Part 1 solves the critical engineering challenges of ROCKET's inference speed and non-differentiability. However, this high-performance implementation is not an end in itself, but rather an enabling tool. It allows us to address a more fundamental scientific question: can we improve upon the "random ensemble" paradigm that sits at the core of the ROCKET model? Having introduced this GPU-accelerated version which replicates the performance of the original ROCKET model, we now use it for all experiments in the second part of this chapter.

While powerful, the standard ROCKET model relies on a brute-force approach, using 10,000 randomly generated kernels in the hope that a sufficiently diverse and powerful feature set will emerge. This process does not address a fundamental limitation: the kernels are static, not specialised to the target dataset, and the ensemble is likely to contain significant redundancy. This section hypothesises that a smaller, more compact set of kernels, if selected

intelligently, could lead to a model that is not only more computationally efficient at inference but also potentially more accurate, as the noise from irrelevant features is removed.

Building on the differentiable framework from Part 1, we can take this concept a step further. Instead of merely selecting a better subset of random kernels, we can optimise them. This section, therefore, investigates a novel methodology for improving both the efficiency and accuracy of ROCKET through a two-stage process. First, we apply state-of-the-art pruning techniques to identify and remove redundant kernels. Second, we unfreeze the weights of the remaining, most salient kernels and fine-tune them end-to-end, transforming them from random initialisations into specialised feature extractors. The goal is to produce a final model that is superior to the original, achieving higher accuracy with a fraction of the computational cost.

7.3.1 Methodology: Pruning and Fine-Tuning Pipeline

The methodology for creating a smaller, more accurate ROCKET model is a two-stage pipeline. The first stage, pruning, applies an established feature selection technique to identify and remove the least important random kernels. Fine-tuning all 10,000 kernels of the original ROCKET model is computationally infeasible, making this pruning stage a necessary prerequisite. The second stage, fine-tuning, takes this smaller, pruned set of kernels and trains their weights end-to-end using the differentiable framework established in Part 1.

7.3.1.1 Kernel Pruning via Sequential Feature Detachment (SFD)

For the pruning stage, we adopt the Sequential Feature Detachment (SFD) methodology, as proposed in the Detach-ROCKET paper [119]. SFD is an iterative, backward elimination algorithm. The process begins with the full set of 20,000 features generated by the 10,000 kernels in the ROCKET model. In each step, a linear Ridge model is trained on the current set of active features. The model's coefficients are used as a proxy for feature importance; features with the smallest absolute coefficient values are deemed least important and are permanently removed. This process of training and removing a fixed percentage of the least important features is repeated until a predefined target number of features remains.

It should be clarified that the SFD method was not developed as part of this thesis. While our initial, unpublished work on this topic involved a similar pruning approach based on Ridge Regressor coefficients, we ultimately adopted the formalised SFD methodology for the work presented in this thesis. This decision was made because the SFD method, as proposed in the

'Detach-ROCKET' paper [119], provided a more robust technique and our primary goal was to enable the kernel fine-tuning investigation by creating a compact kernel set.

7.3.1.2 Pruning Objective: Justifying a Feature-Wise Approach

The primary goal of pruning is to reduce the number of convolutional kernels, as these are the main source of computational cost during both inference and backpropagation. However, the chosen pruning algorithm, Sequential Feature Detachment (SFD), operates on features, not kernels. In the standard ROCKET model, each kernel produces two features (MAX and PPV). This creates a disconnect: pruning a single feature does not remove the underlying kernel, as the kernel remains active to compute its other feature. A kernel is only truly eliminated when all of its associated features have been pruned.

We therefore conducted a preliminary analysis across the 119 datasets from the UCR archive to determine the most effective strategy for maximising the number of eliminated kernels. Two approaches were compared. The first is the standard feature-wise SFD, which prunes the least important features irrespective of their parent kernel. The second is a custom kernel-wise approach, where the importance of a kernel is defined by the summed importance of its features, and entire kernels are pruned as a single unit .

Our analysis revealed that the standard feature-wise approach was significantly more effective. On average, it successfully eliminated **89.72%** of all kernels, leaving an average of 1028 active kernels with at least one remaining feature. Of these, 883 retained only a single feature while 145 retained both. The analysis also revealed that PPV features were retained more often than MAX features (780 vs 393 on average).

In contrast, the kernel-wise approach only eliminated **88.96%** of kernels. Furthermore, the superior pruning efficiency of the standard approach did not come at a greater cost to accuracy. The feature-wise method resulted in an average accuracy degradation of only **-0.62%**, whereas the kernel-wise method led to a more significant drop of **-1.99%**. Given that the standard feature-wise approach is superior on both fronts, it was selected as the definitive pruning strategy and used for all subsequent experiments in this chapter.

7.3.1.3 End-to-End Kernel Fine-Tuning

Following the SFD pruning stage, the final model architecture consists of a trainable convolutional layer containing the pruned subset of kernels, with only filters that have at least one

feature remaining being included in the model, followed by a linear head for classification. The final stage of the pipeline unfreezes the weights of these remaining kernels and fine-tunes them. The core of the fine-tuning investigation is a comparison of two strategies for initialising the weights of this linear head before end-to-end training commences:

- **Fine-Tuning with Ridge Initialisation (FT Ridge Init):** First, a standard ROCKET transform is performed using the pruned, fixed-weight kernels to generate a static feature set from the training data. An 'sklearn' Ridge Classifier is then trained on these features. The learned coefficients from this external classifier are extracted and used to set the initial weights of the final linear layer within the PyTorch model.
- **Fine-Tuning with Random Initialisation (FT Random Init):** This process is identical, except the weights of the PyTorch linear layer are set using the default random initialisation, without leveraging any information from a pre-trained classifier.

After the linear head is initialised using one of these two strategies, the entire PyTorch model—comprising both the pruned convolutional kernels and the linear head—is trained end-to-end. All model parameters are made trainable from the beginning; there is no intermediate freezing stage. The model is trained for 100 epochs using the AdamW optimiser with a learning rate of 0.001. A linear learning rate warmup is applied for the first 5 epochs, followed by a cosine annealing schedule for the remainder of training. The model state corresponding to the epoch with the best validation accuracy is retained for the final evaluation on the test set.

7.3.2 Experimental Setup

7.3.2.1 Method for Handling Failed Experimental Runs

During the experimental evaluation, certain model configurations exhibited numerical instability on specific challenging datasets, resulting in failed runs. A simple mean accuracy calculation across only the successful runs would be misleading, as it would fail to penalise less robust models and would be calculated over an inconsistent number of data points. To address this, a worst-case imputation strategy was adopted. This method treats a failed run not as missing data, but as a catastrophic outcome where the model's performance is considered equivalent to that of a random-chance classifier. This ensures that models that are less stable are appropriately penalised in the final performance summary.

For each failed run on a given dataset, an accuracy score was imputed based on the probability of a random guess being correct. This score is calculated as:

$$Accuracy_{imputed} = \frac{1}{C}$$

where C is the total number of classes in that specific dataset.

This approach is more principled than imputing a static value like zero, as it appropriately scales the penalty relative to the dataset's difficulty. A failure on a binary classification task (where random chance is 50%) is correctly penalised less severely than a failure on a ten-class task (where random chance is 10%).

By imputing failures with a random-chance baseline, the subsequent calculation of the mean accuracy yields a single, robust metric that holistically represents both the predictive power and stability of a model.

Table 7.2: List of sporadic, non-'Symbols' dataset failures handled by imputation. All variants with $p = 1.0$ demonstrated the same failures, while all variants with $p = 4.0$ also demonstrated the same failures, while being different from those of the $p = 1.0$ variants

Model Group	Failed Datasets
$p = 1.0$ variants	Chinatown (2 runs)
$p = 4.0$ variants	GunPointAgeSpan (1 run), GunPointMaleVersusFemale (1 run)

A full list of these infrequent failures is provided in Table 7.2. In addition to these failures, the 'Symbols' dataset was found to have a data-splitting incompatibility with the experimental framework and was therefore excluded from the analysis for all models.

7.3.2.2 Models

The performance of the prune-and-fine-tune methodology is evaluated on **119** datasets from the UCR time series archive. The comprehensive set of models compared in this experiment is detailed in Table 7.3.

7.3.3 Results: Pruning and Fine-Tuning Performance

7.3.3.1 Overall Performance Analysis

The overall mean accuracy for all model configurations, computed across 595 runs (119 datasets with 5 random seeds each) is presented in Table 7.4. The results are sorted by mean accuracy, with failures imputed as described previously.

Table 7.3: Model configurations evaluated in the Part 2 experiments.

Category	Model Configurations
Unpruned Baselines	Unpruned Heaviside Unpruned softPPV (p={1.0, 4.0, 5.0, 10.0})
Pruned-Only Baselines	Pruned Heaviside Pruned softPPV (p={1.0, 4.0, 5.0, 10.0})
Fine-Tuned Models (FT)	Fine-Tuned Ridge Init (p={1.0, 4.0, 5.0, 10.0}) Fine-Tuned Random Init (p={1.0, 4.0, 5.0, 10.0})

Table 7.4: Overall accuracy summary across 119 UCR datasets. Models are sorted by mean accuracy. Failures have been imputed with a random-chance baseline. The 'Numerical Failures Count' column refers to the total number of runs which failed out of the 595 total runs for each configuration (119 datasets $\times$ 5 seeds).

Model Configuration	Mean Acc. (%)	Std Dev	Numerical Failures Count
Unpruned Heaviside	**85.33**	13.98	0
Unpruned softPPV (p=10.0)	84.79	14.14	0
Pruned softPPV (p=10.0)	84.63	14.19	0
Unpruned softPPV (p=5.0)	84.58	14.31	0
Pruned softPPV (p=5.0)	84.42	14.31	0
Unpruned softPPV (p=4.0)	84.37	14.45	2
Pruned Heaviside	84.28	14.48	0
Pruned softPPV (p=4.0)	84.04	14.51	2
FT Ridge Init (p=10.0)	83.51	14.92	0
FT Ridge Init (p=5.0)	83.51	14.82	0
Unpruned softPPV (p=1.0)	83.38	14.61	2
FT Ridge Init (p=4.0)	83.21	14.92	2
Pruned softPPV (p=1.0)	82.80	14.58	2
FT Ridge Init (p=1.0)	82.13	15.02	2
FT Random Init (p=10.0)	77.04	15.73	0
FT Random Init (p=5.0)	76.98	16.09	0
FT Random Init (p=4.0)	76.77	15.74	2
FT Random Init (p=1.0)	75.65	15.90	2

The primary finding from this analysis is that the baseline **'Unpruned Heaviside' ROCKET remains the top-performing model overall**, achieving a mean accuracy of 85.33%. Contrary to our initial hypothesis, the fine-tuning process did not lead to a state-of-the-art result, with all fine-tuned ('FT') models performing worse than the unpruned and pruned-only baselines. However, the results reveal several crucial insights into the behaviour

of the prune-and-fine-tune pipeline, which are explored in the following sections.

7.3.3.2 Pairwise Win/Loss Analysis

While mean accuracy provides a high-level summary, it can be skewed by performance on datasets with different scales of difficulty. A more robust method for comparing classifiers is a direct pairwise comparison across the dataset suite. For each of the 119 datasets, every model is compared against every other model based on their mean accuracy for that specific dataset. Table 7.5 summarises these encounters for key models, showing the total number of datasets on which each model achieved a higher accuracy (a "win").

Table 7.5: Pairwise win-loss-tie (W-L-T) summary for key models across 119 datasets. Each cell shows the record for the row model when compared against the column model. Model abbreviations are: Unpruned Heaviside (U-HV), Unpruned softPPV p=10.0 (U-sPPV), Pruned Heaviside (P-HV), Pruned softPPV p=10.0 (P-sPPV), Fine-Tuned Ridge Init p=5.0 (FT-Ridge), and Fine-Tuned Random Init p=10.0 (FT-Rand).

vs.	U-HV	U-sPPV	P-HV	P-sPPV	FT-Ridge	FT-Rand
U-HV	—	71-35-13	91-22-6	85-27-7	90-21-8	108-11-0
U-sPPV	35-71-13	—	73-43-3	69-43-7	81-35-3	106-13-0
P-HV	22-91-6	43-73-3	—	45-66-8	68-47-4	101-18-0
P-sPPV	27-85-7	43-69-7	66-45-8	—	74-36-9	105-13-1
FT-Ridge	21-90-8	35-81-3	47-68-4	36-74-9	—	91-27-1
FT-Rand	11-108-0	13-106-0	18-101-0	13-105-1	27-91-1	—

The pairwise comparison, detailed in Table 7.5, provides a more granular view of model performance. The results confirm the ranking observed in the mean accuracy analysis and highlight a clear performance hierarchy. The baseline ‘Unpruned Heaviside’ model remains dominant, achieving a strong winning record against every other model. Below the top-tier unpruned models, the pruned baselines show competitive performance, with ‘Pruned softPPV (p=10.0)’ outperforming ‘Pruned Heaviside’ (66-45-8). An interesting interaction emerges from this comparison: while the standard ‘Unpruned Heaviside’ model outperforms its ‘softPPV’ counterpart (71-35-13), this relationship inverts after pruning. This may suggest that the subset of features and kernels that remains after the pruning process benefits somewhat from the soft approximation of the pooling function. This was also seen on our dataset in Figure 7.3, however without pruning. The hierarchy continues downwards, with all pruned models significantly outperforming the fine-tuned variants. The Ridge-initialised fine-tuning strategy (‘FT Ridge

Init') proves far superior to random initialisation ('FT Random Init'), winning on 91 datasets. However, even the best fine-tuned model is clearly outperformed by the simpler pruned-only baselines. This analysis provides strong evidence that, for this collection of datasets, the end-to-end fine-tuning process not only fails to provide an improvement but is actively detrimental to performance. Nevertheless, further exploration of these results can reveal important insights into the weakness of this approach.

7.3.3.3 Statistical Significance and Model Rankings

The Nemenyi post-hoc test is another method of analysing results. This test compares the average rank of each classifier across all 595 experimental runs to determine if their differences are statistically significant. The results are visualised in a Critical Difference (CD) diagram in Figure 7.12. Models that are not statistically significantly different from one another are connected by a horizontal black bar.

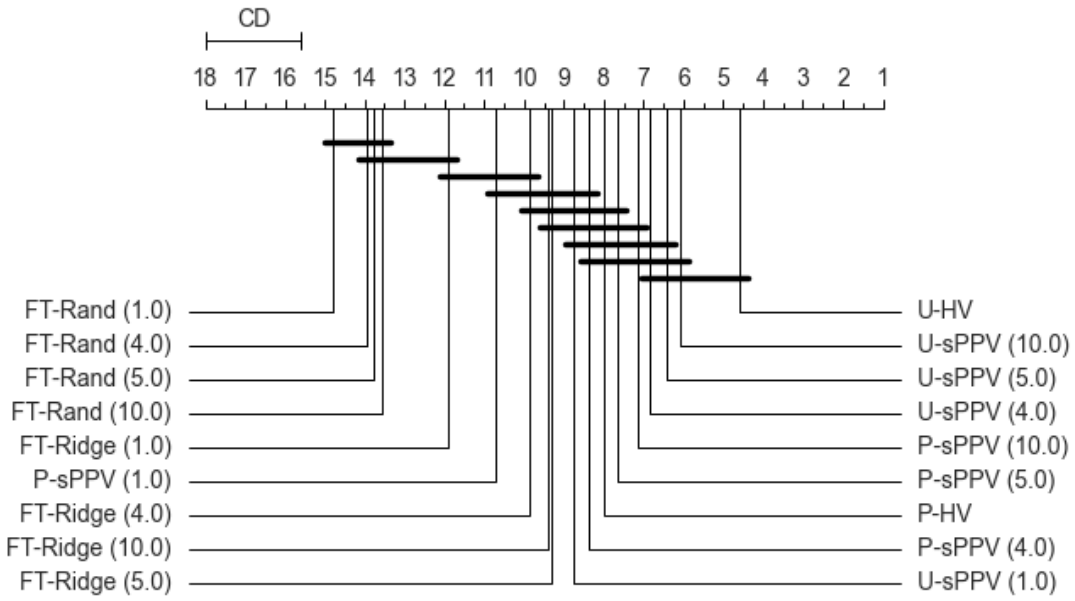


Figure 7.12: Critical Difference diagram comparing all model configurations. A lower rank (further to the right) indicates better overall performance. Models connected by a solid bar are not statistically significantly different at $\alpha = 0.05$.

The CD diagram confirms the findings from the previous analyses. The 'Unpruned Heaviside' model achieves the best (lowest) mean rank of 4.59. A large group of unpruned and

pruned-only models are statistically indistinguishable from one another, forming a clear top tier. In contrast, the fine-tuned models fall into lower-performing tiers. Notably, all 'FT Random Init' models are clustered together as the statistically significantly worst-performing group, highlighting that training the linear head from scratch is a worse initialisation strategy.

7.3.3.4 Ablation Studies and Key Findings

The Effect of Pruning: Comparison with the Detach-ROCKET Paper This part of our study serves as a replication of the pruning methodology from the Detach-ROCKET paper, since we apply the SFD method to the same UCR benchmark datasets. A key objective is to improve efficiency through pruning, and this comparison helps validate our implementation against the original authors' findings.

The original authors report that applying SFD to prune a standard ROCKET model down to 10% of its original features results in a slight performance gain, with an average accuracy improvement of 0.2% across the datasets studied. Our replication shows a comparable, though slightly different, outcome. In our experiments, the 'Unpruned Heaviside' model achieves a mean accuracy of 85.33%, while our 'Pruned Heaviside' model (pruned to a similar 10% level) achieves 84.28%. This represents a drop of approximately 1%, a result which our statistical analysis in Section 7.3.3.3 shows is statistically insignificant. Both the original paper's findings and our own robustly support the primary conclusion: it is possible to remove approximately 90% of ROCKET's kernels with only a minor performance trade-off, leading to a much more efficient model for inference.

The Impact of Linear Layer Initialisation The most striking result from the fine-tuning experiments is that fine-tuning most often degrades performance, revealing a critical insight into the training dynamics. However, the outcome is dataset-dependent, with some notable successes. The 'FT Ridge Init' models, which start with an already strong set of weights in their linear layer, still perform worse on average than their non-fine-tuned 'Pruned' counterparts. This suggests that the end-to-end backpropagation process is less effective at finding an optimal solution than simply training a linear model on the static features. The degradation is even more severe for the 'FT Random Init' models, which perform worst of all.

A closer look at individual datasets reveals specific cases where fine-tuning was beneficial. For the 'FT Ridge Init' model, significant accuracy gains were observed on datasets such as 'Beef' (+12.67%) and 'DodgerLoopWeekend' (+7.14%). These successes suggest that on

certain data distributions, end-to-end training can specialise the random kernels effectively. Conversely, the process was highly detrimental on other datasets, particularly those with a large number of classes like 'GestureMidAirD2' (-10.62%) and 'GestureMidAirD3' (-9.69%), where performance dropped significantly. The 'FT Random Init' models exhibit even more volatile performance, suffering catastrophic drops in accuracy of up to 40% on small datasets like 'BME'.

This indicates that the loss landscape is difficult to navigate. While fine-tuning offers potential upside on a small subset of problems, it is not a universally beneficial strategy for ROCKET's features, and for the majority of datasets in this collection, it harms overall performance.

Table 7.6: Mean accuracy comparison for models grouped by the 'softPPV' steepness parameter (p). Higher values for p consistently yield better results.

	Mean Accuracy (%) for p-value			
Model Type	**1.0**	**4.0**	**5.0**	**10.0**
Unpruned softPPV	83.22	84.21	84.58	**84.79**
Pruned softPPV	82.63	83.87	84.42	**84.63**
FT Ridge Init	81.96	83.04	**83.51**	**83.51**
FT Random Init	75.49	76.60	76.98	**77.04**
Overall Average	80.82	81.93	82.37	**82.49**

The Impact of the `softPPV` Parameter For all models using the differentiable 'softPPV' function, there is a clear and consistent trend related to the steepness parameter, p (referred to as λ in the methodology). As shown in Table 7.6, higher values of p lead to better performance across all model categories, not fine-tuned and fine-tuned alike. A value of $p = 10.0$, which corresponds to a steeper sigmoid that more closely approximates the true Heaviside function, is consistently among the best-performing options, although it is noteworthy that for the 'FT Ridge Init' models, the $p = 5.0$ variant achieves an identical mean accuracy and wins the direct pairwise comparison. This suggests that while differentiability is necessary for training, an activation function that behaves more decisively like a step function is beneficial for average accuracy. This preference may be particularly pronounced across the UCR archive, as many of its datasets have a limited number of training instances, making them difficult to train.

A less flexible, more rigid activation could act as a form of regularisation, preventing the model from deviating from its robust initial state. This finding is particularly revealing for the

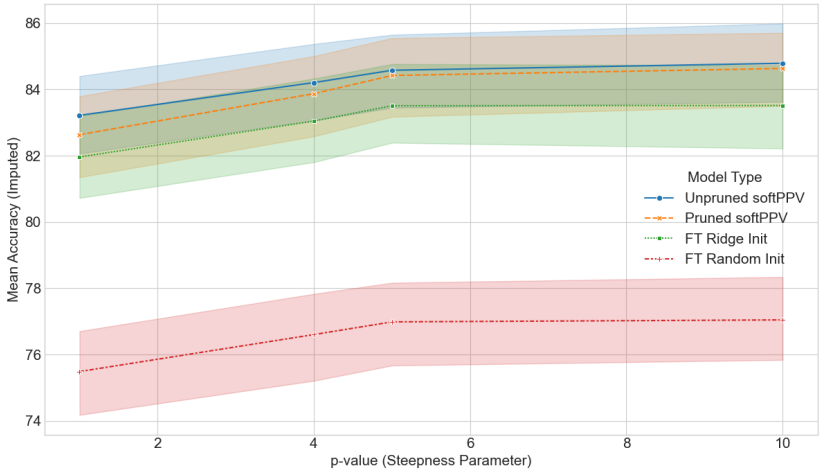


Figure 7.13: Impact of the softPPV steepness parameter (p) on model performance. The plot shows the mean accuracy across all 119 UCR datasets for each major model category as the steepness parameter increases. Error bars represent the standard deviation of accuracy across the datasets. A clear trend is visible where higher values of p, which more closely approximate the non-differentiable Heaviside function, consistently result in higher mean accuracy for all model types.

fine-tuning experiments. A larger steepness parameter creates a sigmoid function with near-flat, saturated regions where the gradient is close to zero. This saturation severely restricts the flow of gradients during backpropagation, hindering the model's ability to learn from the data. The fact that the best-performing fine-tuned models are precisely those with the most limited trainability is a strong indication that the gradient-based updates were not effective. Performance is highest when the softPPV function most closely mimics the original, non-differentiable Heaviside function, and the influence of fine-tuning is minimised. Furthermore, it is plausible that the initialisation with highly effective pruned features places the model in a strong local minimum; subsequent gradient-based updates may then struggle to find a better solution and instead risk moving towards a worse one, therefore leading to a preference for a closer to Heaviside PPV.

7.3.3.5 Deep Dive: The Impact of Dataset Characteristics on Fine-Tuning

To better understand the conditions under which fine-tuning succeeds or fails, a deeper analysis is performed by correlating model performance against key dataset characteristics. This analysis first explores the weaknesses of the randomly initialised fine-tuning approach compared to a static baseline, before directly comparing the two fine-tuning initialisation strategies against each other. For this analysis, we report the Spearman rank correlation coefficient (ρ), as it

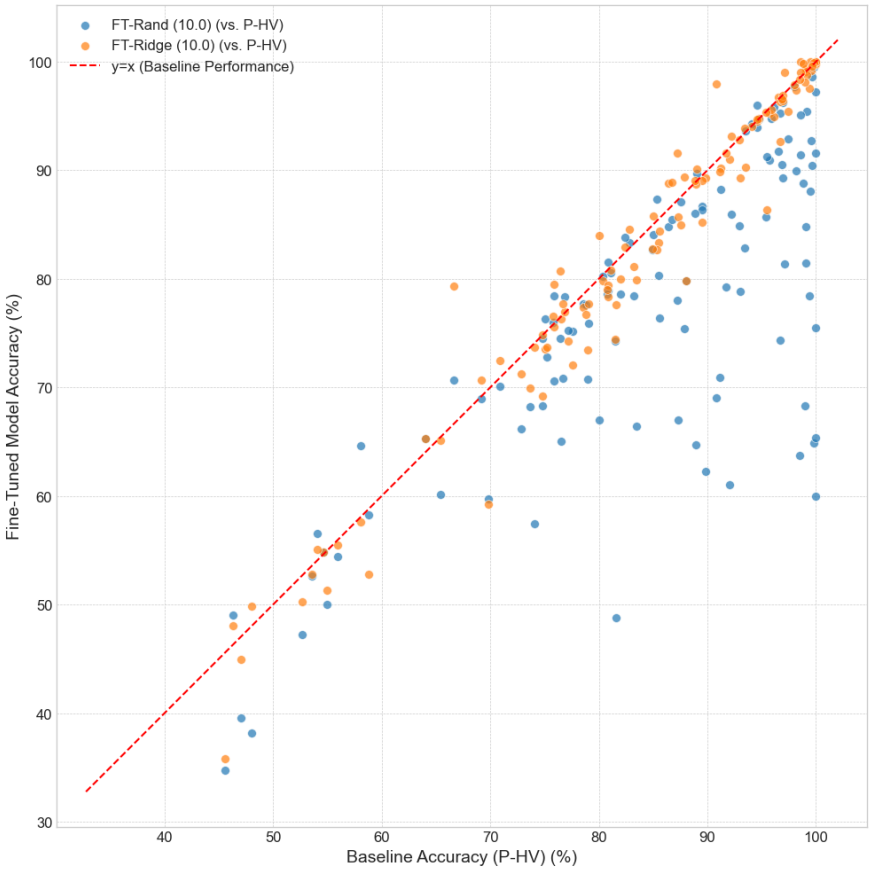
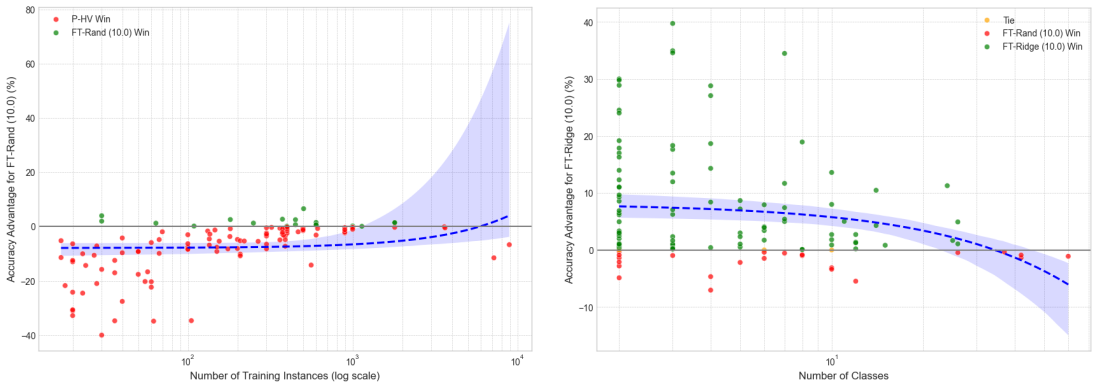


Figure 7.14: Per-dataset comparison of the two fine-tuning strategies against the Pruned Heaviside (P-HV) baseline. Each point represents one of the 119 datasets. The plot visually demonstrates the instability of the randomly initialised model (FT-Rand), which exhibits many catastrophic failures (points far below the diagonal line), compared to the more robust Ridge-initialised model (FT-Ridge), whose performance clusters more tightly around the baseline.

is more robust to the non-linear relationships and outliers expected across the diverse UCR archive. Table 7.7 summarises the results, with significant correlations ($p < 0.05$) shown in bold.

Fine-Tuning Performance vs. Baselines The analysis reveals distinctly different behaviours for the two fine-tuning strategies when compared against the 'Pruned Heaviside' (P-HV) baseline, as shown in Table 7.7.

The performance of the 'FT-Rand' model is strongly dependent on the size of the training set. We found a strong, statistically significant positive correlation between its performance advantage over the baseline and the number of training instances ($\rho = 0.6653, p < 0.0001$).



A. FT Random Init (p=10.0) Advantage vs. Pruned Heaviside

B. FT Ridge Init (p=10.0) Advantage vs. FT Random Init

Figure 7.15: Visual analysis of model performance advantage versus dataset characteristics. Each point represents a UCR dataset. The y-axis shows the accuracy difference; positive values indicate the first-named model won. **(a)** Demonstrates the strong dependency of the 'FT Random Init' model on the number of training instances. **(b)** Shows the consistent advantage of 'FT Ridge Init' over 'FT Random Init', which has a statistically significant negative correlation with the number of classes in the dataset.

Table 7.7: Spearman rank correlation (ρ) between model performance advantage and dataset characteristics. Performance advantage is calculated as the accuracy of the first-named model minus the second. For all fine-tuned (FT) and softPPV (sPPV) models shown, the 'p=10.0' variant is used. Significant p-values ($p < 0.05$) are shown in bold.

	Training Instances		Time Series Length		Number of Classes	
Comparison	ρ	**p-value**	ρ	**p-value**	ρ	**p-value**
FT-Rand vs. P-HV	0.6653	**< 0.0001**	0.1174	0.2034	0.0250	0.7872
FT-Ridge vs. P-HV	-0.1265	0.1703	-0.2133	**0.0198**	-0.4258	**< 0.0001**
FT-Ridge vs. FT-Rand	-0.6887	**< 0.0001**	-0.2108	**0.0214**	-0.2233	**0.0146**
P-HV vs. U-HV	0.0867	0.3483	-0.0694	0.4532	-0.0740	0.4237
U-sPPV vs. U-HV	0.0662	0.4746	-0.0506	0.5850	0.0799	0.3879
P-sPPV vs. P-HV	0.0124	0.8934	-0.1177	0.2025	-0.1872	**0.0414**

This confirms the visual trend in Figure 7.15a: the randomly initialised model performs catastrophically poorly on datasets with few training examples, but its performance improves as the number of samples increases. However, the model typically becomes 'less bad' rather than superior, rarely surpassing the non-fine-tuned baseline. This suggests that while a large dataset is a necessary condition for a randomly initialised model to learn meaningful weights, the fine-tuning process struggles to converge to a better solution than the original static, random features combined with a ridge classifier.

The data in Table 7.9 strongly reinforces this conclusion. The model's five largest losses

all occur on datasets with very few training instances (between 20 and 105), with the model's worst performance being a 40% accuracy drop on 'BME', a dataset with very few samples. This confirms that the random initialisation strategy is completely unsuitable for small-data regimes.

In stark contrast, the 'FT-Ridge' model's performance shows no significant correlation with the number of training instances. Its pre-initialised linear layer provides a strong starting point that makes it more robust on smaller datasets. However, its performance is significantly and negatively correlated with dataset complexity, specifically time series length ($\rho = -0.2133, p = 0.0198$) and, more strongly, the number of classes ($\rho = -0.4258, p < 0.0001$). This indicates that while Ridge initialisation is an improvement over random starting weights, the fine-tuning process is in most cases on this dataset collection detrimental, but especially so on longer or more complex tasks (i.e., more classes). This is exemplified in Table 7.8, where its largest performance losses occur on datasets with many classes, such as 'GestureMidAirD2' and 'GestureMidAirD3'.

Directly comparing the two initialisation strategies, the **'FT Ridge Init'** model consistently outperforms the 'FT Random Init' model. This performance gap is also correlated with dataset characteristics; the advantage of Ridge initialisation is significantly larger on datasets with fewer training instances ($\rho = -0.69, p < 0.001$) and fewer classes ($\rho = -0.22, p = 0.015$).

Comparing Initialisation Strategies A direct comparison between the two fine-tuning strategies (FT-Ridge vs. FT-Rand) further clarifies the role of initialisation. The results in Table 7.7 show a strong, significant negative correlation between the performance advantage of 'FT-Ridge' and the number of training instances ($\rho = -0.6887, p < 0.0001$), as well as weaker negative correlations with time series length ($\rho = -0.2108, p = 0.0214$) and the number of classes ($\rho = -0.2233, p = 0.0146$). The correlation with training instances is the key finding: it demonstrates that the benefit of using Ridge initialisation over random initialisation is most pronounced on datasets with fewer training instances. As the amount of data increases, the 'FT-Rand' model begins to catch up, but for small-data problems, a good initialisation is critical.

Exploratory Findings The broader analysis in Table 7.7 revealed few other strong trends. The effects of pruning alone ('P-HV vs. U-HV') and the use of softPPV in the unpruned state ('U-sPPV vs. U-HV') showed no significant correlation with any of the measured dataset characteristics. This suggests their impact on performance is relatively consistent across different

dataset sizes and complexities. The only other significant, albeit weak, correlation was found for 'P-sPPV vs. P-HV', whose performance advantage has a slight negative correlation with the number of classes ($\rho = -0.1872, p = 0.0414$).

Table 7.8: Top 5 performance gains and losses for the 'FT Ridge Init' model compared to the 'Pruned Heaviside' baseline.

Largest Performance Gains (Top 5 Wins)				
Dataset	**Acc. Advantage (%)**	**Instances**	**Timestamps**	**Classes**
Beef	+12.67	30	470	5
DodgerLoopWeekend	+7.14	18	288	2
ShakeGestureWiimoteZ	+4.40	50	385	10
ChlorineConcentration	+4.31	467	166	3
BeetleFly	+4.00	20	512	2
Largest Performance Losses (Top 5 Losses)				
Dataset	**Acc. Advantage (%)**	**Instances**	**Timestamps**	**Classes**
GestureMidAirD2	-10.62	208	360	26
GestureMidAirD3	-9.69	208	360	26
HouseTwenty	-9.08	40	2000	20
ACSF1	-8.20	100	1460	10
CricketX	-7.03	390	300	12

Table 7.9: Top 5 performance gains and losses for the 'FT Random Init' model compared to the 'Pruned Heaviside' baseline.

Largest Performance Gains (Top 5 Wins)				
Dataset	**Acc. Advantage (%)**	**Instances**	**Timestamps**	**Classes**
EthanolLevel	+6.60	504	1751	4
Beef	+4.00	30	470	5
ScreenType	+2.72	375	720	3
WormsTwoClass	+2.60	181	900	2
SemgHandMovementCh2	+2.53	450	1500	6
Largest Performance Losses (Top 5 Losses)				
Dataset	**Acc. Advantage (%)**	**Instances**	**Timestamps**	**Classes**
BME	-40.00	30	128	3
InsectEPGRegularTrain	-34.86	62	600	10
UMD	-34.72	36	150	3
Plane	-34.67	105	144	7
Rock	-32.80	20	2844	4

7.4 Conclusion

In conclusion, this chapter has presented a two-part investigation into enhancing the ROCKET model. In Part 1, we addressed the critical engineering challenge of inference speed by developing a GPU-accelerated PyTorch implementation of the ROCKET transform. This framework was shown to provide a significant speed-up (approximately 690%) over existing CPU-based implementations, making it a viable tool for rapid prediction tasks such as on-line monitoring in steel manufacturing. Our benchmarks demonstrate clear crossover points where the GPU's parallel processing becomes advantageous. On the univariate UCR datasets, the GPU implementation surpasses the standard CPU version on datasets with more than approximately 4,000 samples. The synthetic data analysis provides a more detailed picture, revealing the critical impact of data dimensionality on this crossover point. For a 20-channel time series, for instance, our GPU implementation becomes faster than its CPU counterpart with as few as 2,656 timesteps, and for data with 50 or more channels, the GPU is faster across almost all configurations. The same trend holds for the number of samples: while a single-channel series of 1,000 timesteps requires over 3,200 samples for the GPU to be faster, a 20-channel series of the same length needs only 88 samples before the GPU takes the lead. A key innovation is the novel differentiable 'softPPV' function, which maintains predictive accuracy while enabling the end-to-end training of kernel weights.

Building on this framework, Part 2 investigated a methodology for improving efficiency through a process of pruning and fine-tuning. The main findings were threefold. First, by applying the SFD algorithm from the work by Uribarri et al. [119], we confirmed that feature-wise pruning is highly effective, capable of removing ~90% of ROCKET's kernels with only a minor, statistically insignificant drop in accuracy. Second, we showed that the strategy used to initialise the final linear layer is critical, with a Ridge-initialised model significantly outperforming a randomly-initialised one. Finally, and most surprisingly, we found that end-to-end fine-tuning of the pruned kernels consistently degraded performance compared to using the static, pruned random features.

This investigation contributes a powerful new tool to the time series community in the form of a fast, differentiable ROCKET implementation. Furthermore, it provides a rigorous analysis of a prune-and-fine-tune methodology, offering valuable insights into its practical limitations

and demonstrating that for ROCKET, a simpler, pruned approach is more effective than a more complex, fine-tuned one.

Chapter 8

Prediction of Roughness Parameter R_a

Having explored methods for reconstructing missing data in steel surface profiles and a novel approach to improve the performance of the family of ROCKET machine learning models, we now turn our attention in this chapter to our main problem of interest of the project and an item of great interest in steel production: the direct prediction of the surface roughness parameter, R_a.

As established in detail in Chapter 1, the core motivation for this work stems from the challenge of controlling surface roughness during temper rolling. For high-value products, line operators—or a potential future automated system—must adjust parameters like roll force and speed to meet customer requirements. However, the current feedback process is slow, relying on stylus measurements taken from small sections at the head or tail of a coil, which may not be representative of the entire surface and prohibits mid-coil adjustments [25]. This delay causes issues with just-in-time manufacturing and can force costly re-production cycles [107]. An accurate, on-line monitoring system that provides measurements for the entire surface is therefore essential. Such a system would not only empower operators with immediate feedback but also opens the door for the real-time, closed-loop control systems needed to fully automate the temper mill.

The on-line measurement technique used by our industrial partner fires a laser at the steel surface and captures the scattered light intensities at various angles. In theory, surface gradients can be calculated from these reflections, integrated to form a surface profile, and then used to derive surface statistics [26]. However, this physics-based, closed-form solution has proven insufficiently accurate for production release when compared to traditional stylus-based methods, creating the central challenge this thesis aims to solve.

This thesis investigates two distinct machine learning philosophies to address this challenge, as illustrated conceptually in Figure 8.1. The first, explored in Chapter 6, is a multi-stage "reconstruction-first" pipeline. This approach uses machine learning to identify and repair flawed sections of a calculated surface profile, with the goal of producing a high-fidelity profile from which any surface statistic (including R_a, defined in Section 2.2.2) can be calculated. While this method is versatile, its success depends on the performance of multiple sequential steps, such as the accuracy of the initial closed-form calculation, the gap-identification function, and laser measurement, even in perceived high-quality regions.

In contrast, this chapter investigates a conceptually simpler, "end-to-end" or "black-box" approach. The hypothesis here is that a sufficiently powerful machine learning model can learn the complex transformation from raw laser reflection data directly to the final R_a value, accounting for unknown factors that influence the physical processes and raw reflections such as light being reflected away. This method is less interpretable but may be more robust by learning a direct mapping without accumulating errors from a multi-stage process, ultimately allowing the model to transform the sensor data to align with the gold-standard stylus measurement. This is subtly different from the closed-form calculation performed on the laser measurement data, which can only act as a proxy for the gold standard, offering one more potential benefit of using machine learning to directly map to the stylus measurements.

In this chapter, we formulate the manufacturing issue into a Time Series Extrinsic Regression problem and a Machine Vision problem and leverage state-of-the-art machine learning models to enhance the transformation of on-line measurements into a significantly more accurate R_a surface roughness metric. By comparing a selection of data-driven approaches, including both deep learning such as convolutional, recurrent, and transformer networks and non-deep learning methods such as ROCKET and XGBoost, to the closed-form transformation, we evaluate their potential using Root Mean Squared Error (RMSE) and correlation for improving surface texture control in temper strip steel manufacturing.

Our experiments are performed using the labelled on-line laser-based surface dataset described in Section 4.3.1. We not only demonstrate the potential of machine learning in very accurately solving this complex transformation task but also provide a methodology and evaluation of these models which others can use for similar problems. Our results showed that the 2D xresnet34 model offered the greatest improvement over the closed-form baseline approach, while the 1D convolution-based TCN model also performed well while offering the benefit of superior computational efficiency. This chapter is based on work published in the Springer

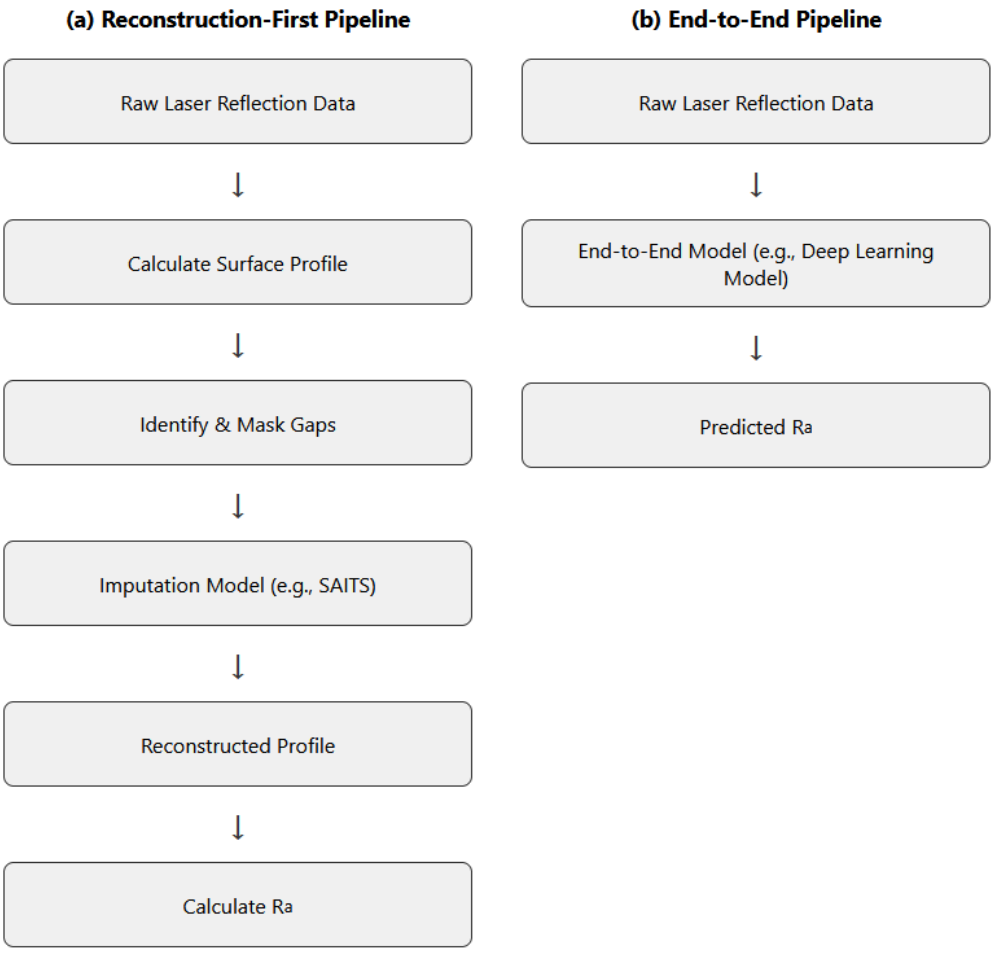


Figure 8.1: A conceptual diagram comparing the two machine learning pipelines investigated in this thesis. (a) The multi-stage "reconstruction-first" pipeline from Chapter 6, which aims to create a high-fidelity surface profile for subsequent analysis. (b) The "end-to-end" pipeline investigated in this chapter, which learns a direct mapping from raw sensor data to the final R_a statistic, bypassing intermediate steps.

International Journal of Advanced Manufacturing Technology [85].

8.1 Methodology

We apply non-deep learning and deep learning models from different areas of research in order to judge their suitability for the problem of R_a prediction in terms of both accuracy and speed. These approaches will be compared to the baseline obtained from calculating an R_a from a

profile built from gradients.

The experiments in this chapter are conducted using the labelled steel surface dataset. The complete details of the data acquisition process, the dataset's structure, and the "many-to-many" labelling strategy are described in Chapter 4. The following sections detail the specific components of this methodology.

As the data is 2-dimensional data, we include deep learning models commonly used for image classification. Classification models can be modified to produce a regression output by changing the output layer of the model. We also experiment with the top 1-dimensional deep learning time series classification models, treating each of the individual sensors as a separate channel. We note that these models are designed with data that has a long time step dimension in mind, whereas the 2-dimensional models typically come from machine vision and expect a square input image with a substantially shorter length dimension than ours of 65,536. 1-dimensional models also have the potential to be faster due to lower computational complexity.

All data-driven models are benchmarked against the R_a from the proprietary, physics-based "closed-form" solution provided by the commercial measurement device itself. As detailed in Chapter 4, this method involves a multi-step process: calculating surface gradients from the raw light intensities, integrating them to form a surface profile, filtering out low-frequency waviness, and finally calculating the R_a statistic. However, our own replication of this process does not perfectly match the device's output, revealing significant discrepancies for outlier samples (as illustrated in Figure 8.2). These differences are likely due to proprietary algorithms for handling the data gaps and signal noise shown in Figure 8.3. Therefore, to ensure the most rigorous and industrially-relevant comparison, the baseline for all results is the direct R_a value produced by the commercial device, not our own calculated version.

A plot of raw intensity data can be seen in Fig. 8.3 in raw unprocessed form, and with thresholding to remove sensor issues at very low intensities. The worst outliers have lots of gaps in the intensity data which is likely what is causing the discrepancies in our calculations. We believe that the measurement device might have a better method for interpolating the raw data than the one we have used. It is not in the scope of our work to re-engineer the closed-form solution exactly as the measurement device calculates it, we just provide here an insight into the process. Therefore, we use the values from the measurement device as the closed-form baseline values and not our calculated values.

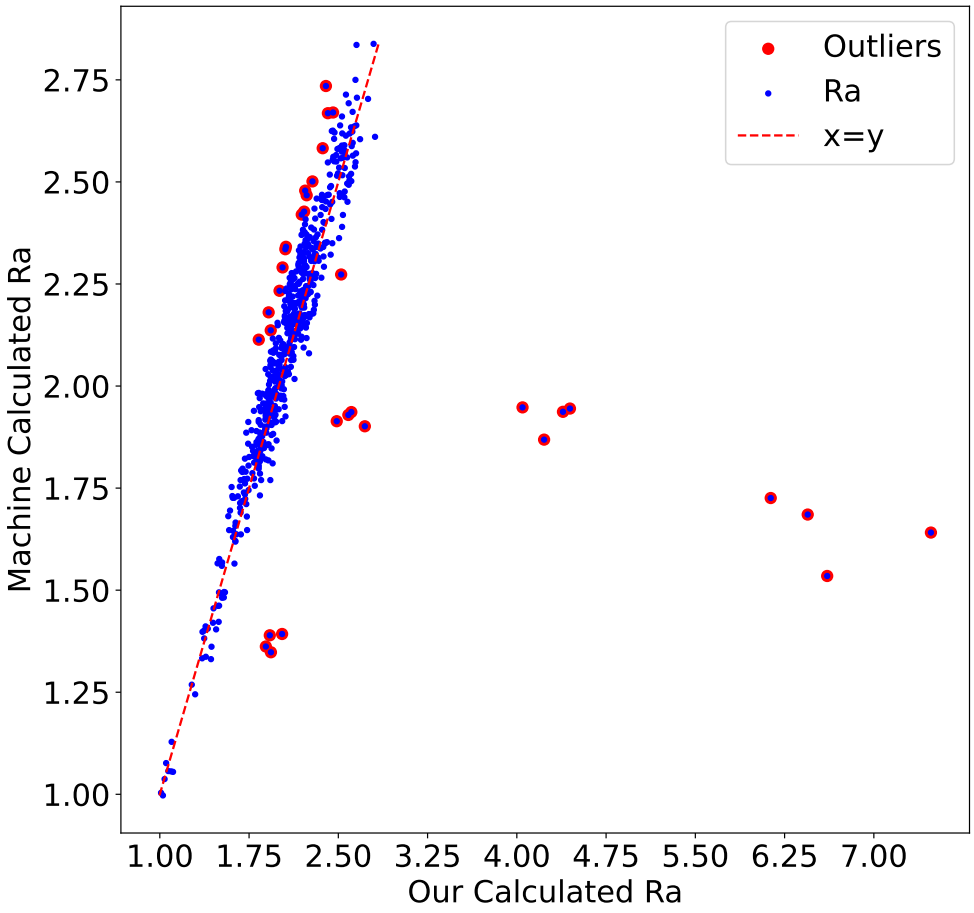
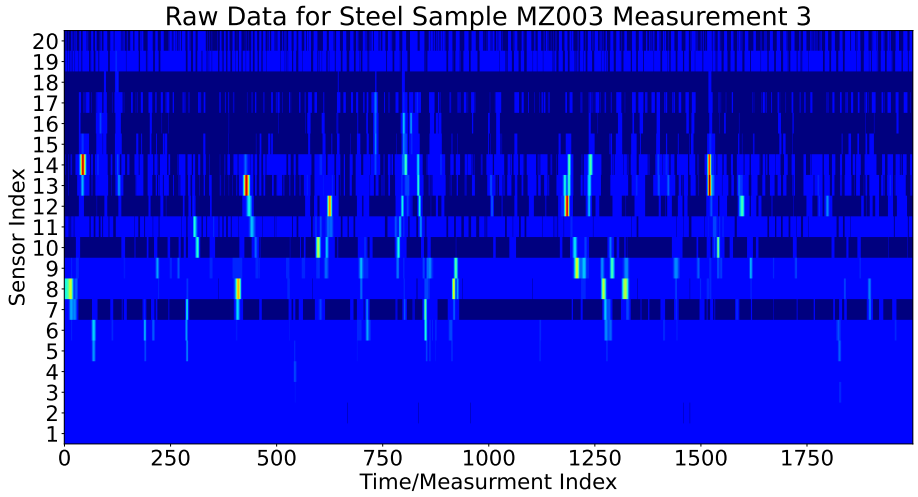


Figure 8.2: The difference between the R_a calculated from the raw laser-reflected data by us and by the measurement device itself.

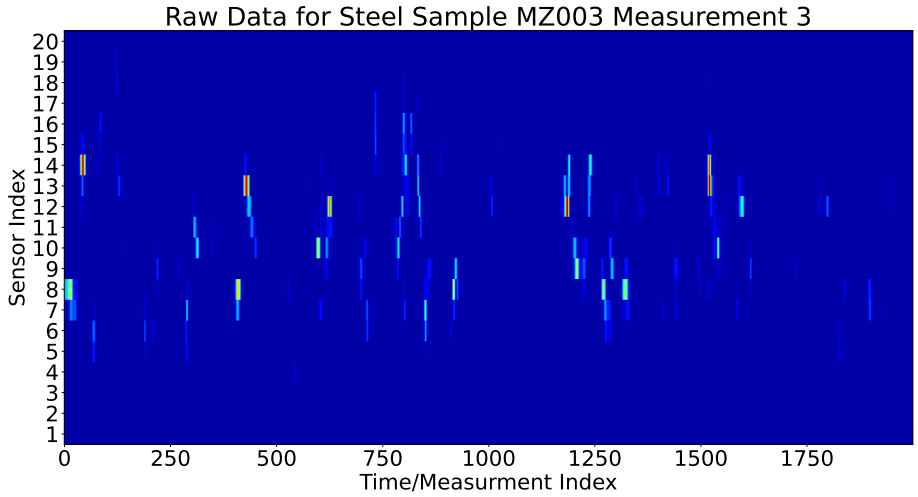
8.2 Models tested

To find the most effective architecture for this direct prediction task, we benchmarked a diverse suite of models. These can be grouped into three main categories: classical non-deep learning models, 1D deep learning models designed for time series, and 2D deep learning models adapted from computer vision. The following sections provide a technical description of each model tested.

Crucially, as highlighted in Chapter 3, there are no pre-existing deep learning architectures



A. No Thresholding



B. With Thresholding

Figure 8.3: Comparison of raw intensity data for the worst outlier with and without thresholding.

tailored for this specific application or for similar high aspect ratio data. Consequently, there is no single 'state-of-the-art' model to adopt and refine. Instead, we must conduct a broad, first-principles investigation to determine the fundamental nature of the data: is it best treated as a multivariate time series (prioritising temporal dynamics along the strip) or as a textural image (prioritising spatial coherence across sensors)? Furthermore, we must establish whether the high training computational cost of deep feature learning is justifiable against industrial

latency constraints, or if random feature transformations (like ROCKET) provide sufficient predictive performance. The selection of models below is specifically designed to cover these competing hypotheses.

8.2.1 Non-Deep Learning Models

As state-of-the-art non-deep learning models for time series tasks, ROCKET and its faster variant MiniROCKET are benchmarked. These models operate not as end-to-end learners, but as highly efficient feature extractors. The core mechanism involves applying thousands of fixed, non-trainable convolutional kernels—each with randomly initialised parameters like weight, length, and dilation—to the input time series. From the output of each convolution, summary statistics such as the maximum value (max pooling) and the proportion of positive values (PPV) are extracted to form a large feature vector. This vector is then used to train a simple and fast linear model, such as a ridge regressor. A full description is provided in Section 2.5.1. Due to the speedup gained, we make use of our custom PyTorch implementation of the ROCKET model described in Chapter 7, which runs on a GPU.

This chapter also evaluates classical tree-based ensembles. A decision tree is a non-parametric model which learns a set of rules to partition the feature space to which the data belongs. At any node, the tree searches for the optimal split, defined by a feature j and a threshold s, that minimises the residual sum of squares.

The first model we make use of which is based on decision trees is the Random Forest, introduced by Breiman [14]. This is a powerful ensemble learning algorithm that combines multiple decision trees for accurate predictions. It handles high-dimensional data, reduces overfitting, and provides insights into feature importance. With its versatility, efficiency, and interpretability, Random Forest is widely used in various domains and applications. The algorithm works by constructing a multitude of decision trees on randomly selected subsets of the training data, created by drawing n samples from the original training data with replacement. Each tree in the Random Forest operates independently, trained on one of these subsets. In addition to its unique data subset, each tree randomly selects a subset of m features to consider for each split, with $m \approx \sqrt{p}$, where p is the total number of features. This decorrelates the tree from others in the random forest. The final prediction of the Random Forest is the average of the predictions from all trees. This ensemble approach helps to reduce overfitting and improves the overall robustness of the model. Random Forest has demonstrated its effectiveness across various domains and applications, including classification, regression, and feature selection.

XGBoost, introduced by Chen and Guestrin [18], is a widely used machine learning algorithm which combines gradient boosting and tree-based models to handle diverse data types and capture complex interactions. XGBoost sequentially adds decision trees to a growing ensemble, where each subsequent tree corrects the errors made by the previous trees. The prediction at step t is the prediction from the previous step, plus the output of a new decision tree f_t:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$$

At each step, XGBoost seeks to add the tree f_t that minimises the regularised objective function:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

$$\text{Obj}^{(t)} = \sum_{i=1}^{n} L\big(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\big) + \Omega(f_t)$$

where L is a differentiable loss function, such as mean squared error in the case of regression, and Ω is a regularisation term that penalises complexity. For even better computational efficiency and tractable optimisation, XGBoost makes use of a second-order Taylor expansion of the loss function to simplify this objective for the new decision tree to:

Using a second-order Taylor expansion around $\hat{y}_i^{(t-1)}$:

$$L(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) \approx L(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2$$

where $g_i = \partial_{\hat{y}_i^{(t-1)}} L(y_i, \hat{y}_i^{(t-1)})$ and $h_i = \partial^2_{\hat{y}_i^{(t-1)}} L(y_i, \hat{y}_i^{(t-1)})$ are the first and second derivatives of the loss evaluated at $\hat{y}_i^{(t-1)}$.

Substituting this approximation into $\text{Obj}^{(t)}$ and discarding the constant term $L(y_i, \hat{y}_i^{(t-1)})$, which does not depend on f_t, and noting that $f_t(x_i) = w_j$ for all $x_i \in I_j$, the objective can be rewritten at the leaf level as:

$$\text{Obj}^{(t)} \approx \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T$$

where T is the number of leaves in the decision tree, I_j is the set of data points in leaf j, w_j is the weight for leaf j, g_i and h_i are the first and second order gradients of the loss function, and λ and γ are regularisation parameters.

XGBoost can additionally support various objective and loss functions and the regularisation helps to prevent overfitting and enhance generalisation. Its exceptional performance, versatility, and scalability have made it a popular choice in competitions and real-world applications.

Table 8.1: Overview and rationale for the machine learning model categories benchmarked.

Category	Models Benchmarked	Architectural Rationale
Non-Deep Learning	**ROCKET / MiniROCKET:** Random convolutional kernel transforms. **Tree-Based Ensembles:** Random Forest and XGBoost.	Serve as strong, interpretable baselines and are generally fast to train. May not capture complex hierarchical features as effectively as deep models.
1D Deep Learning	**Inception Family:** InceptionTime, XceptionTime. **FCN-RNN Hybrids:** MRNN-FCN, etc. **ResNet & Variants:** ResNet, xResNet (1D). **Foundational Baselines:** FCN, 1D-ResNet (Wang et al.), ResCNN.	Computationally efficient and specifically designed for long temporal patterns. Treat sensor readings as independent channels, ignoring vertical spatial relationships.
2D Deep Learning	**xResNet (2D Variants):** 2D version of the ResNet architecture. **Modern ConvNets:** ConvNeXt. **Vision Transformers:** ViT, Swin v2.	Leverage the 2D image-like nature of the data to learn spatial features across sensors. Incur a significant computational cost and must handle an extreme aspect ratio.

8.2.2 1D Deep Learning Models

The first category of deep learning architectures benchmarked are those designed to operate on 1D sequential data, with many coming from the Time Series Extrinsic Regression (TSER) [113] and Time Series Classification (TSC) [58] areas of research. These models are specifically designed to operate on 1D sequential data with long time step dimensions, which aligns with our data characteristics. In our approach, we treat the data from each of the 20 sensors as individual input channels. The models are grouped below by their core architectural family.

InceptionTime InceptionTime, proposed by Fawaz et al. [59], adapts the Inception module [112] from computer vision for TSC. It consists of an ensemble of deep neural networks.

Each network uses multiple Inception blocks, which apply convolutional filters of different lengths (e.g., 10, 20, 40) in parallel to the same input. This allows the model to capture patterns at multiple temporal scales simultaneously. The outputs are concatenated, passed through residual connections, and finally a Global Average Pooling (GAP) layer.

In each of the blocks, the model applies a bottleneck layer to the input, which reduces the dimensionality of the data. Subsequently, it employs 10, 20, and 40-length convolutional filters within the blocks. Additionally, an alternative path applies a max pooling layer to the input, followed by a bottleneck layer. To obtain a global representation of the features, a Global Average Pooling (GAP) layer is employed. Finally, the architecture concludes with a linear prediction layer which can be used for classification or regression.

Building on this, Rahimian et al. propose XceptionTime [99] as a dedicated model for analyzing surface Electromyography (sEMG) signal data. The authors draw inspiration from the success of two existing models: Xception, known for its superior performance in large image classification compared to Inception V3, and InceptionTime, which was developed based on the Inception V4 concept. Motivated by these achievements, the authors introduce XceptionTime as a novel model that combines the strengths of InceptionTime with the utilisation of depthwise separable convolution Xception modules. By incorporating depthwise separable convolutions, XceptionTime outperforms its existing counterparts in analyzing sEMG signal data.

FCN-RNN Hybrids The MRNN-FCN model architecture, proposed by Khan et. al. [64], incorporates two parallel feature extraction routes, which are subsequently concatenated in the final feature layer. Path 1, begins with a dimension shuffle operation, followed by the selected RNN unit and dropout regularisation for training purposes.

Path 2 encompasses a combination of different layers. It begins with a 1D convolutional layer with Batch normalisation and ReLU activation. Subsequently, a squeeze-and-excitation block is employed, which is a channel-wise attention mechanism that adaptively re-weights each feature map based on its global importance, allowing the model to focus on the most informative channels. Additional convolutional layers and squeeze-and-excitation blocks follow. Finally, a GAP operation is used to obtain a summary representation of the extracted features.

The outputs from Path 1 and Path 2 are concatenated, facilitating the fusion of diverse features derived from different pathways, and providing a comprehensive representation of the input data.

The MRNN-FCN, MGRU-FCN, and MLSTM-FCN models share a similar architecture, with the key difference lying in the type of RNN module utilised. While the MRNN-FCN employs a standard RNN module, the MGRU-FCN and MLSTM-FCN models replace it with the Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) modules, respectively. The GRU and LSTM modules are variants of the RNN module that address the vanishing gradient problem and enable the model to capture long-term dependencies more effectively.

ResNet and Variants He et al. [52] introduced the ResNet framework to facilitate the training of significantly deeper networks compared to previous approaches. This framework has revolutionised the fields of computer vision and deep learning, as residual learning and skip connections have become integral components in many modern deep architectures. Residual learning, in combination with skip connections, represents the key innovation of ResNet. It enables the network to bypass multiple layers and directly propagate information from earlier layers to subsequent ones, thereby ensuring a smooth flow of gradients during training and mitigating the vanishing gradient problem.

The ResNet framework is structured as a sequence of layers, with each layer comprising residual blocks. These blocks consist of multiple convolutional layers, batch normalisation, Rectified Linear Unit (ReLU) activation, and a shortcut connection that adds the input of the block to its output. The architecture is designed to accommodate deeper networks, typically achieved by increasing the number of channels while reducing spatial dimensions. This is accomplished by downsampling and progressively augmenting the number of channels in each subsequent layer. This architectural strategy empowers ResNet models to effectively capture and process complex features at varying scales and depths, leading to enhanced performance in diverse computer vision tasks.

A more recent evolution of the ResNet architecture is xResNet. He et al. [53] proposed a series of modifications, referred to as a “bag of tricks", that can be applied to the ResNet architecture, resulting in variants known as xresnets. While xresnets maintain the core principles of ResNet, utilising residual blocks with convolutional layers and shortcut connections, xresnets incorporate notable structural changes. In particular, xresnets modify the input stem by splitting the original convolutional layer with a kernel size of 7 and a stride of 2 into three separate convolutional layers. Each of these layers utilises a kernel size of 3. The first layer employs a stride of 2, while the subsequent two layers have a stride of 1. Additionally, xresnets modify the downsampling of the identity path. Instead of employing a kernel size of 1 with a stride of 2, xresnets utilise an average pooling layer with a kernel size of 2, followed by a

kernel size 1 convolutional layer with step size of 1. Furthermore, xresnets introduce an expansion parameter, which, when not a value of 1, adds an additional convolutional layer to each block and augments the number of channels. The 1D variants used in our experiments include xresnet1d18, xresnet1d34, xresnet1d50, xresnet1d101, and some of their 'deep' and 'deeper' versions.

Foundational Deep Learning Baselines from TSC In addition to the more recent architectures, we also benchmark several foundational deep learning models that are commonly used as baselines in TSC literature.

In our experiments, we have utilised the ResNet model proposed by Wang et al. [121] specifically designed for 1D time series data. This implementation deviates from conventional ResNet architectures in several aspects. Firstly, it employs 1D convolutions instead of 2D convolutions, thereby accommodating the nature of time series data. Unlike traditional ResNet models, this variant does not incorporate downsampling operations. However, it still increases the number of channels across layers to capture diverse temporal patterns. Moreover, this model diverges in terms of its depth, comprising only three layers as opposed to the standard ResNet architecture with four layers. Furthermore, it introduces a decreasing kernel size pattern across the three layers, transitioning from a kernel size of 7, to 5, and finally to 3. Typically, ResNet architectures employ kernel sizes of 3 throughout all layers, except for the input layer.

From the same authors (Wang et al.), we also test the simpler Fully Convolutional Network (FCN) [121]. The FCN is a straightforward model composed of three consecutive 1D convolutional blocks. These blocks extract feature channels in a sequence of 128, 256, and 128, respectively. Following each convolutional block, batch normalisation and ReLU activation functions are applied. To obtain a global representation, global average pooling is employed. The final prediction is made using a linear layer.

The ResCNN network proposed by Zou et. al. [135] is another ResNet-inspired architecture composed of four sequential blocks. The first block features two paths: the first path includes three convolutional blocks, each incorporating batch normalisation and ReLU activation, while the second path consists of a single convolutional block with batch normalisation and no ReLU activation. The output of the second path acts as a shortcut and is added to the output of the first path, followed by a ReLU activation. Subsequently, each block performs 1D convolution, followed by batch normalisation and an activation function. The first block employs leaky ReLU activation, the second block utilises Parametric ReLU (PReLU) activation, and the final block applies Exponential Linear Unit (ELU) activation. To capture a global

representation of the feature maps, global average pooling is performed. Finally, a linear layer is used for prediction.

8.2.3 2D Deep Learning Models

For the 2D deep learning approach, we explore models from the machine vision field, as our data has a 2D structure where the sensor channels are spatially related. A key challenge is that machine vision models typically expect square-shaped images, whereas our data forms long, thin arrays. To adapt these models, we treated each sample as a single-channel grayscale image with a height of 20 (representing the sensors) and a width of 2^{16}. Due to the high computational cost of this approach, a more selective suite of architectures is benchmarked.

xResNet (2D Variants) The 2D versions of xResNet, specifically xresnet18 and xresnet34, are also benchmarked. These models share the same architecture described in the 1D ResNet and Variants section, but employ 2D convolutional filters to operate on the data as an image.

ConvNeXt ConvNeXt, introduced by Liu et al. [78] in the era dominated by transformer architectures, offers a compelling alternative to traditional ConvNets. While Vision Transformers (ViTs) quickly surpassed ConvNets as the state-of-the-art image classification model, they encountered challenges when applied to general computer vision tasks. Hierarchical Transformers, like Swin Transformers, reintroduced ConvNet priors, making Transformers more applicable to various vision tasks. The authors incorporate design elements from Vision Transformers into a standard ResNet architecture to unleash the potential of pure ConvNets. The result is ConvNeXt, a family of models constructed entirely from standard ConvNet modules. ConvNeXt achieves remarkable accuracy and scalability, outperforming Swin Transformers on multiple benchmarks, while retaining the simplicity and efficiency of ConvNets. These findings challenge prevailing beliefs and emphasise the importance of convolution in computer vision. Architecturally, ConvNeXt adapts a standard ResNet by incorporating several modern techniques. It heavily utilises depthwise separable convolutions arranged within an inverted bottleneck block. Key changes inspired by Vision Transformers include using larger 7x7 kernel sizes, replacing ReLU with the GELU activation function, and favouring Layer Normalisation over Batch Normalisation. Two variants are used in our experiments: Tiny and Small.

Vision Transformer (ViT) The Vision Transformer (ViT), introduced by Dosovitskiy et al. [34], represents a significant breakthrough in computer vision by leveraging transformer-

based architectures, which have excelled in natural language tasks. ViT challenges the conventional wisdom that Convolutional Neural Networks (CNNs) are not necessary and a pure transformer can perform very well for image classification. The self-attention mechanisms of transformers, applied to embedded patches, capture global contextual information from input images, enabling the model to comprehend intricate visual patterns and relationships. The self-attention mechanism facilitates efficient computation of pairwise interactions between image patches, allowing ViT to model long-range dependencies effectively. ViT directly applies a pure transformer encoder architecture to sequences of embedded image patches. The ViT architecture consists of an initial patch embedding layer, which splits the input image into a sequence of fixed-size patches. These patches are then linearly projected into a set of learnable embeddings. The resulting embeddings are augmented with positional encodings to encode spatial information. The transformer encoder layers process the embedded patches, incorporating self-attention and feed-forward neural network modules. The self-attention mechanism captures the global context by attending to all image patches, enabling effective information exchange. Finally, a classification token is added, and the resulting sequence is passed through a linear layer followed by softmax activation to obtain the class probabilities. We have changed the model slightly by adjusting the patch size to 20 by 32 instead of the 16 by 16 patch size in the original model, such that the patches fit the 20 sensor dim exactly and the 32 (2^5) is divisible by the 2^{16} length of the data, resulting in 2048 patches. We experiment with the ViT Tiny and the ViT Small models.

Swin Transformer Swin v2, introduced by Liu et al. [77], improves upon previous transformer-based models which perform global image processing. It adopts a hierarchical processing strategy to effectively handle high-resolution images. Building upon the Swin Transformer architecture, which reintroduces ConvNet priors, Swin v2 serves as a practical choice for a generic vision backbone. By dividing the input image into non-overlapping patches and organising them into stages and windows, the model captures both local and global information efficiently. A key innovation is the use of shifted windows between consecutive self-attention layers, which enables cross-window connections and allows information to propagate across the image without the quadratic cost of global attention. Additional enhancements, such as the patch merging module and shift operation, contribute to improved feature extraction and contextual modelling. In our experiments, we evaluate the performance of the Swin v2 Tiny and Swin v2 Small variants. These models are chosen based on their proven performance in various machine learning tasks and their potential to improve the accuracy of laser reflection

measurements for on-line sheet steel manufacturing.

8.3 Experimental Setup

All deep learning experiments are conducted using the PyTorch framework. For the implementation of specific model architectures, we utilise several established libraries. For the implementation of the 1D deep learning models, we utilised the tsai package [93]. Meanwhile, we modified several 2D models from the TorchVision package [80] to suit our specific data format and experimental requirements, and use the fastai package for the xresnet models [55].

For each deep learning experiment, we employ a consistent training loop and set of hyperparameters. The chosen optimiser is Lion [24], with a weight decay of 0.01. To assess performance during training, the Mean Squared Error (MSE) loss function (defined in Section 2.3.2) is used by the optimiser. The initial learning rate is set to 0.0001, and a learning rate scheduler using the 'reduce-on-plateau' strategy is utilised. This scheduler reduces the learning rate by a factor of 0.5 when the validation loss fails to improve for 10 epochs. Training continues until the validation loss fails to improve for 100 consecutive epochs. Throughout the training process, checkpoints are saved at regular intervals, and the checkpoint with the best validation loss is selected for the final model.

The hyperparameters for the non-deep learning experiments are as follows. For XGBoost we use a cross-validated grid search with 3 folds. The search parameters are: 100, 500, and 1000 number of estimators, max depth of 5, 10, 15, and 20, and learning rate of 0.1, 0.05, and 0.01. For Random Forest Regressor we also use cross-validated grid search with 3 folds. The search parameters are: number of estimators 100, 500, and 1000, max depth of 5, 10, 15, and 20, and minimum samples per leaf 1, 5, 10, and 15.

The TCN models have kernel size (ks) and a specification for the number of layers (nl) as hyperparameters. The layers hyperparameter contains a list where the length is the number of layers and the contents specify the output channel dimension for the convolution in each of those layers. We use the default value of 25 for each channel dimension and vary the number of layers. The specific ks and nl hyperparameters we use for each model are noted in the results tables. Some models are originally designed for classification but also work very well for regression; for these, a parameter "channels out" is set to 1 since we are regressing to a single value. Some models also have a 'sequence length' parameter, which we set to the length of the data, 2^{16}. Most of the other deep learning models do not have primary hyperparameters

to set, or they have internal optional hyperparameters that are not expected to be changed, and therefore we use the default values.

8.3.1 Training Procedure

Two methods of training are performed. In the first, 20% of the data is held back for the test set in order to evaluate the models' performance. Due to the limited spread of the R_a label values from the many-to-many problem, we perform the split on each steel sample in order to guarantee that the model has seen 80% of the input samples from each steel sample.

In the second method, due to the relatively small dataset size of only 49 different steel samples, we believe that it is important to test the model's generalisation ability on completely unseen new steel samples. This is done by sectioning our data into k folds, where each of the folds contains all measurements from a single steel sample. A new model is trained for each fold where one of the folds is used as the test set and all the other folds as training data.

As the model has not seen this steel sample before, these experiments ensure the model is not simply grouping data samples assigning the related R_a from the same steel samples and instead is actually learning a useful transformation from Raw reflected readings to the R_a.

Before feeding the data into the model, thresholding is performed as specified in Eq. 4.1 and then z-score normalisation. Calculated mean and standard deviation values from the training data are used to normalise the test data.

8.4 Results

We compare our data-driven approaches to the closed-form baseline for the calculation of the R_a from the laser reflection data. Fig. 8.4 illustrates that the closed-form baseline systematically overestimates R_a relative to stylus measurements. In order to improve the baseline results we have transformed the calculated R_a values onto the true $x = y$ line.

8.4.1 20% data split experiments

The first set of conducted experiments are on the data split such that 20% of data samples from each steel sample are used as the test data. The results of the 20% data split analysis are presented in Table 8.2, which provides an overview of the performance metrics for the evaluated models, categorised based on their approach. The analysis includes three approach types: closed-form baselines, non-deep learning (data-driven approaches), and deep learning

Table 8.2: Results on Data: 20% per Sample

Model	RMSE	Correlation	Max Error	Pred. Coverage
Closed-form baseline				
Baseline transformed	0.3076	0.5940	1.0193	0.0992
Baseline	0.6546	0.5940	1.2648	0.0064
Non-deep learning (data-driven approach)				
MiniROCKET	0.0660	0.9576	0.2078	0.5921
Rocket	0.0882	0.9259	0.3876	0.4737
XGBoost	0.1970	0.5431	0.7598	0.2303
Random Forest	0.1985	0.6012	0.7038	0.2105
1D deep learning (data-driven approach)				
TCN ks=9 nl=12	0.0487	0.9770	0.1740	0.6908
TCN ks=5 nl=13	0.0499	0.9762	0.1881	0.6776
TCN ks=7 nl=8	0.0529	0.9729	0.1891	0.6579
MGRU-FCN	0.0564	0.9695	0.1948	0.6053
InceptionTime	0.0584	0.9677	0.2476	0.6447
xresnet1d34	0.0618	0.9639	0.3486	0.5921
MLSTM-FCN	0.0633	0.9609	0.2052	0.5789
xresnet1d18	0.0643	0.9597	0.3837	0.6382
xresnet1d50 deep	0.0644	0.9600	0.3110	0.6118
xresnet1d101	0.0662	0.9587	0.2221	0.5329
MRNN-FCN	0.0703	0.9517	0.3980	0.6513
xresnet1d50 deeper	0.0704	0.9513	0.3141	0.5329
xresnet1d18 deep	0.0712	0.9506	0.4461	0.6053
xresnet1d34 deep	0.0743	0.9478	0.4354	0.5789
xresnet1d50	0.0754	0.9460	0.4752	0.5658
xresnet1d34 deeper	0.0779	0.9405	0.3381	0.5461
xresnet1d18 deeper	0.0795	0.9418	0.5058	0.5921
XceptionTime	0.0896	0.9201	0.3476	0.4474
ResCNN	0.0917	0.9190	0.5634	0.5987
FCN	0.1123	0.8799	0.6776	0.5197
ResNet	0.1223	0.8691	0.8352	0.6382
2D deep learning (data-driven approach)				
xresnet34	0.0453	0.9806	0.1390	0.7237
ConvNeXt Small	0.0554	0.9706	0.1982	0.6184
ConvNeXt Tiny	0.0561	0.9701	0.1796	0.6382
xresnet18	0.0621	0.9629	0.2787	0.5658
Swin v2 Tiny	0.1124	0.8737	0.4015	0.2961
Swin v2 Small	0.1508	0.7551	0.8138	0.2961
ViT Small	0.1729	0.6859	0.4881	0.2632
ViT Tiny	0.1947	0.5813	0.5736	0.2171

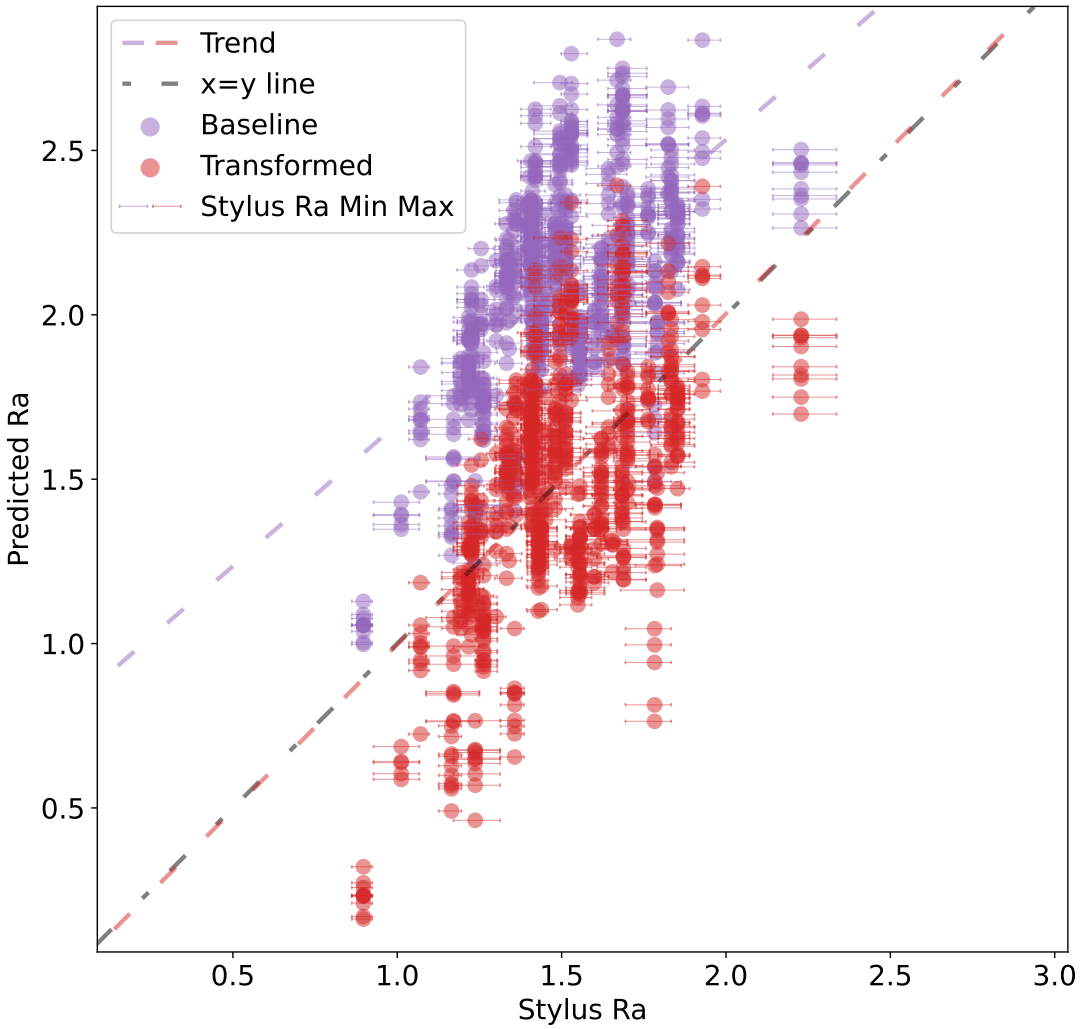


Figure 8.4: The closed-form baseline R_a results and the results transformed onto the $x = y$ line.

(data-driven approaches). The metrics used for evaluation include the Root Mean Squared Error (RMSE) (as defined in Section 2.3.2, Pearson's Correlation, maximum prediction error, and the percentage of predictions within the minimum and maximum values obtained from the stylus measurement for the steel sample. The table is sectioned by type of approach, where each section is in descending order of RMSE with the best result displayed in bold.

The **Closed-Form Baseline** models, including Baseline Transformed and Baseline, served as a reference point for comparison. These models achieved RMSE values of 0.3076 and 0.6546 respectively. Notably, the prediction coverage was relatively low for both models, with values of 0.0992 and 0.0064, respectively, meaning that many results fell outside the range of values

provided by the stylus measurement.

For the **Non-deep learning (data-driven approach)** models, the MiniROCKET model demonstrated the best performance, achieving the lowest RMSE value of 0.0660 and a high correlation coefficient of 0.9576. The ROCKET model achieved a slightly lesser performance, with an RMSE of 0.0882 In contrast, the XGBoost and Random Forest models performed poorly compared to the other models tested, as well as in comparison to the baseline models. For XGBoost and Random Forest models, the input data must be completely flattened, resulting in a very large feature space of 1,310,720 (20x2**16) dimensions for each data sample. We suspect that the results suffer from the large dimensions and would benefit from some dimensionality reduction, such as PCA before data is input into the model.

In the category of **1D deep learning (data-driven approach)**, the models demonstrated strong performance. The TCN ks=9 nl=12 model achieved the lowest RMSE of 0.0487, with a high correlation coefficient of 0.9770. Other models in this category, such as TCN ks=5 nl=13, TCN ks=7 nl=8, MGRU-FCN, and InceptionTime, demonstrated comparable performance with RMSE values ranging from 0.0499 to 0.0584. Additionally, models such as xresnet1d34, MLSTM-FCN, and xresnet1d50 deep exhibited good performance with RMSE values below 0.065 and correlation coefficients above 0.9500. It is interesting that the TCN ks=7 nl=8 performed similarly to the other two models despite this variation not having a receptive field large enough to encompass the entire data length. This might be due to the parameter R_a being, in essence, a mean across the length of the sample, such that the output regression layer can easily combine the features from different regions.

2D Deep Learning (Data-Driven Approach) models showed some variation in performance. It is important to note that the 2D Deep Learning models can be categorised into two groups: convolution-based models (such as xresnet and ConvNeXt) and transformer-based models (such as Swin and ViT). The best-performing model, xresnet34, was better than any of the 1D models, though many of the transformer-based 2D models performed worse than all of them. In the cases where a 2D model corresponded to a 1D architecture, the 2D model performed better. For example, this was the case with the xresnet models. The xresnet34 model achieved the lowest RMSE of 0.0453, with a high correlation coefficient of 0.9806. Similar performance was observed for the ConvNeXt Small and ConvNeXt Tiny models, which achieved RMSE values of 0.0554 and 0.0561 respectively. The xresnet18 model also exhibited relatively strong performance. The transformer-based models exhibited relatively lower performance, as indicated by higher RMSE values and lower correlation coefficients compared to

the convolution-based models.

Overall, the best-performing model is the 2D xresnet34. It exhibited the lowest RMSE, highest correlation coefficient, lowest maximum error, and best prediction coverage. This is closely followed by the best 1D convolution-based TCN model with similar performance but with the benefit in terms of computational efficiency due to less complex 1D convolutional operations.

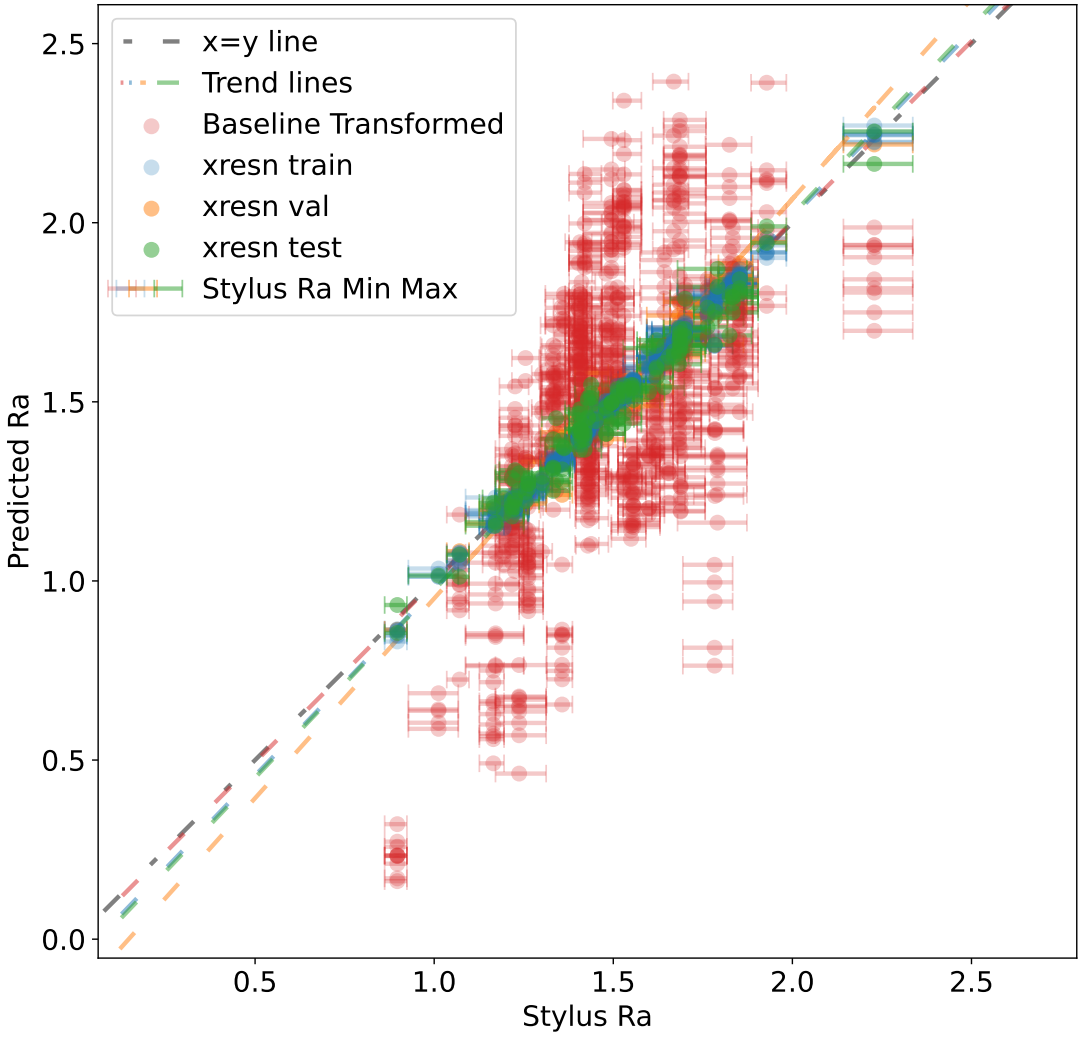


Figure 8.5: The 2D xresnet34 results on the 20% split dataset vs the baseline transformed.

The prediction scatter for the 2D xresnet34 on the 20% data is shown in Fig. 8.5. It is the best deep learning model using the data-driven approach and significantly improves R_a predictions when compared to the transformed closed-form baseline. Prediction coverage (Pred.

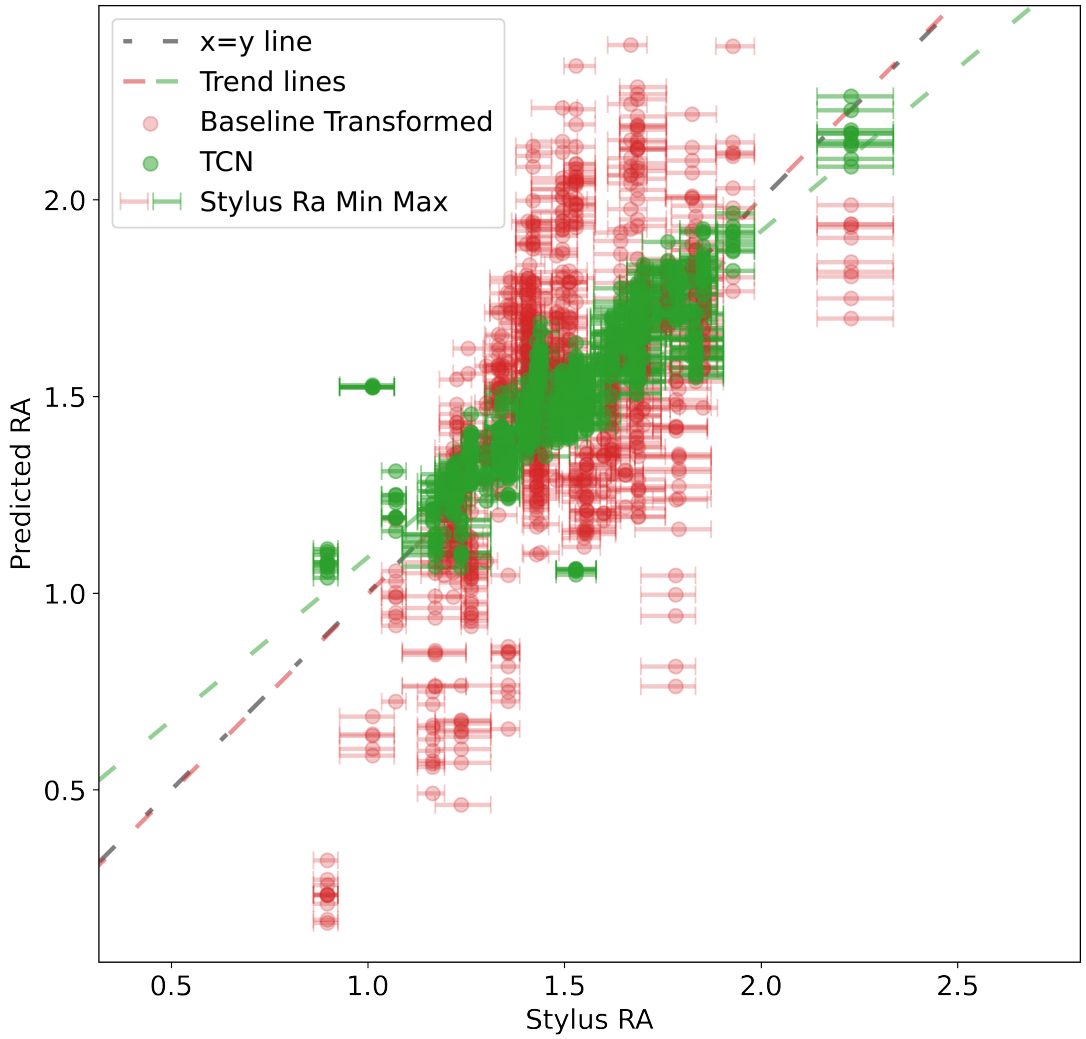


Figure 8.6: The results for the best model on the k-fold data (TCN ks=5 nl=13) vs the baseline.

Coverage), is a measure of the proportion of predictions that are within the minimum and maximum of R_a measurements from the stylus for the steel sample. This is visualised in the scatter plots: when a point's error bars cross the x=y line, it is counted as correct for the coverage percentage.

8.4.2 Leave-One-Out Experiments

To test the models' generalisability between data samples from different steel samples, k-fold experiments are performed where all samples from each steel sample become the test set for

a model trained for each. We alternatively call these "leave-one-out" experiments, as in each case, we have left out the data of one steel sample. The results of the leave-one-out experiments are presented in Table 8.3. These experiments are significantly more expensive to run, as each model needs to be retrained 49 times, once for each steel sample. Therefore, these experiments have fewer models tested. The baselines are the same for both experiments as they are closed form.

Table 8.3: Results on Data: Fold Each Steel Sample

Model	RMSE	Correlation	Max Error	Pred. Coverage
Baselines				
Baseline transformed	0.3076	0.5940	1.0193	0.0992
Baseline	0.6546	0.5940	1.2648	0.0064
Non-deep learning (data-driven approach)				
MiniROCKET	0.1157	0.8675	0.3944	0.3587
Rocket	0.1427	0.7861	0.4852	0.2813
1D deep learning (data-driven approach)				
TCN ks=5 nl=13	0.1028	0.8923	0.5167	0.4542
TCN ks=7 nl=8	0.1036	0.8917	0.4902	0.4361
xresnet1d34	0.1076	0.8819	0.5649	0.4284
xresnet1d18	0.1102	0.8784	0.6172	0.4284
TCN ks=9 nl=12	0.1109	0.8764	0.5412	0.3832
InceptionTime	0.1115	0.8717	0.5904	0.4503
MLSTM-FCN	0.1174	0.8602	0.6120	0.3729
xresnet1d50	0.1204	0.8490	0.6394	0.3768
MGRU-FCN	0.1224	0.8512	0.5105	0.3200
MRNN-FCN	0.1267	0.8365	0.5793	0.3561
FCN	0.1359	0.8041	0.5446	0.3006
2D deep learning (data-driven approach)				
xresnet34	0.1095	0.8774	0.6010	0.4181

The k-fold experiments evaluate the performance of the models on each steel sample individually. In these experiments, the baseline transformed and baseline models serve as the reference baselines for comparison. The baseline transformed model achieves an RMSE value of 0.3076 and a correlation coefficient of 0.5940. The Baseline model performs slightly worse, with an RMSE value of 0.6546 and the same correlation coefficient.

Among the non-deep learning models in the data-driven approach category, the MiniROCKET model demonstrates the best performance in the k-fold experiments. It achieves an RMSE value of 0.1157 and a high correlation coefficient of 0.8675. The ROCKET model also performs reasonably well, with an RMSE of 0.1427 and a correlation coefficient of 0.7861.

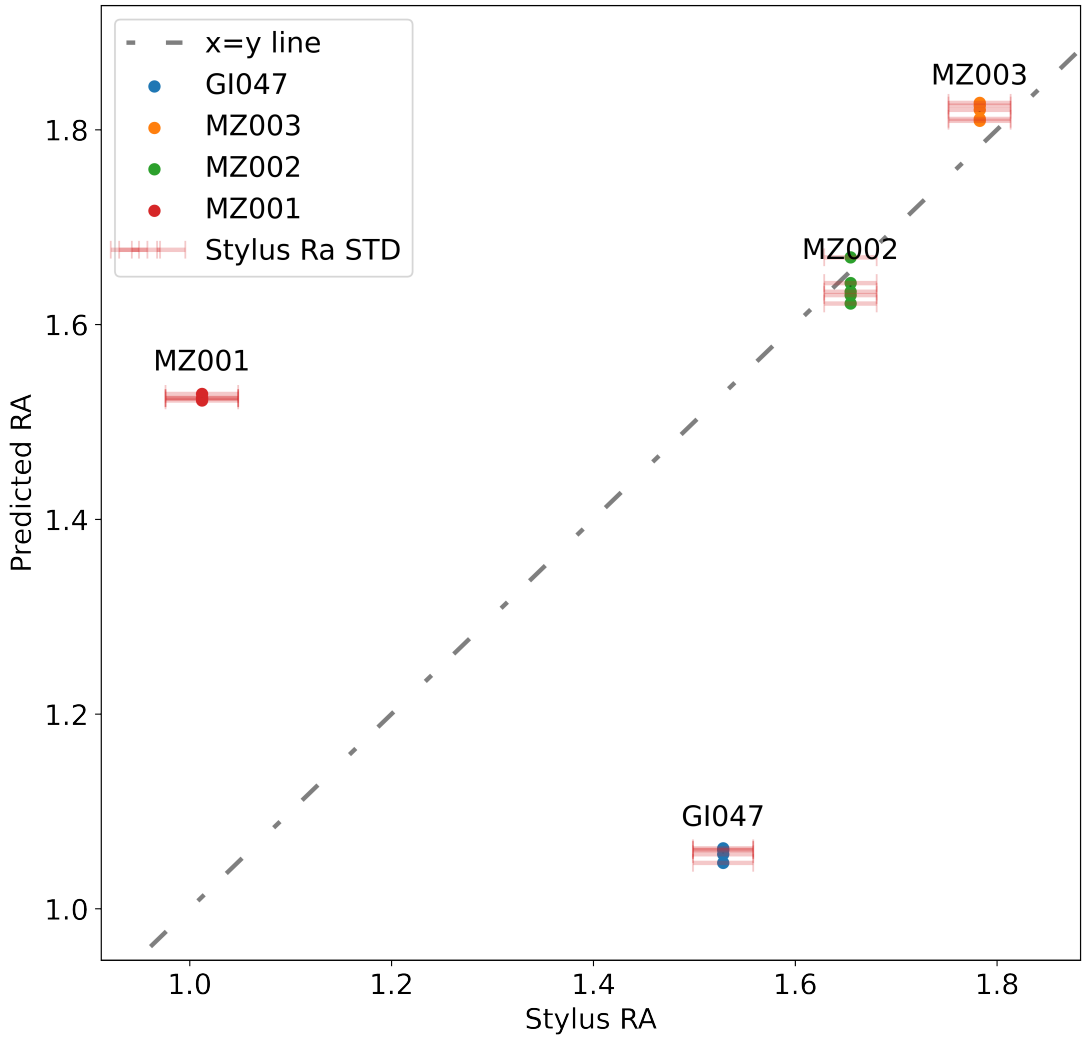


Figure 8.7: The TCN ks=5 nl=13 model results scatter plot for all of the steel samples that were outliers when we calculated the closed-form results, namely, all of the “MZ" steel samples and the “GI047" steel sample.

In the 1D deep learning models, the TCN ks=5 nl=13 and TCN ks=7 nl=8 models exhibit strong performance. Both models achieve relatively low RMSE values of 0.1028 and 0.1036, respectively, with high correlation coefficients of 0.8923 and 0.8917. The 2D xresnet34, achieves an RMSE value of 0.1095 and a correlation coefficient of 0.8774. This is slightly worse than its 1D equivalent, unlike the 20% split results.

Figure 8.6 provides a visual comparison between the predictions of the best-performing k-fold model (TCN ks=5 nl=13) and the closed-form baseline. The TCN predictions align

much more closely with the stylus-measured Ra values, showing a tighter clustering around the $x = y$ line compared to the baseline. In contrast, the baseline exhibits a clear systematic deviation and greater dispersion, particularly at higher Ra values. This visual evidence supports the quantitative improvements reported in Table 8.3, confirming the superior generalisation performance of the data-driven model in the leave-one-out setting.

Comparing the performance of the models in the k-fold experiments, it can be observed that the TCN models, lower capacity 1D xresnet models (xresnet1d34, xresnet1d18) models, and 2D xresnet34 outperform the other models. These models perform well across both experiments, with the TCN models constantly performing very well and very significantly beating the baselines.

As expected, the models perform worse on the k-fold experiments as each prediction is on a completely unseen steel sample, and our dataset has only 49 steel samples, a relatively small number for generalisation across samples.

Fig. 4.9 shows that the data has close to a normal distribution of R_a values across our samples. The model has seen more samples with R_a values closer to the mean across all samples. Fig. 8.8 shows the RMSE for each steel sample for the TCN models for the k-fold evaluation. It plots the results vs the stylus measurement for each of the steel samples. There appears to be a minor improvement in the results nearer to the distribution mean.

It can be observed that in the k-fold experiments, all the models exhibited poor performance on all of the samples taken from two specific steel samples, namely “MZ001” and “GI047”. Interestingly, these two samples were also identified as outliers in our baseline calculations, as depicted in Fig. 8.2. This finding raises the possibility that the models are struggling to accurately predict these results due to inherent issues with the collected data for these samples or problems arising from our data preprocessing procedures. It is important to note that these preprocessing steps are conducted prior to both our calculation of the closed-form solution and the data-driven approach. However, it is worth mentioning that the data-driven approach produced satisfactory results for the other two outliers from the closed-form calculation, namely “MZ002" and “MZ003". Fig. 8.7 shows the TCN ks=5 nl=13 results scatter plot of only the samples from the closed-form outlier steel samples, “MZ001", “MZ002", “MZ003", and “GI047".

Despite these issues, the data-driven approach consistently beats the baseline across all metrics. It is also worth noting that the CNN models are computationally efficient and can be run in real-time for predictions during production.

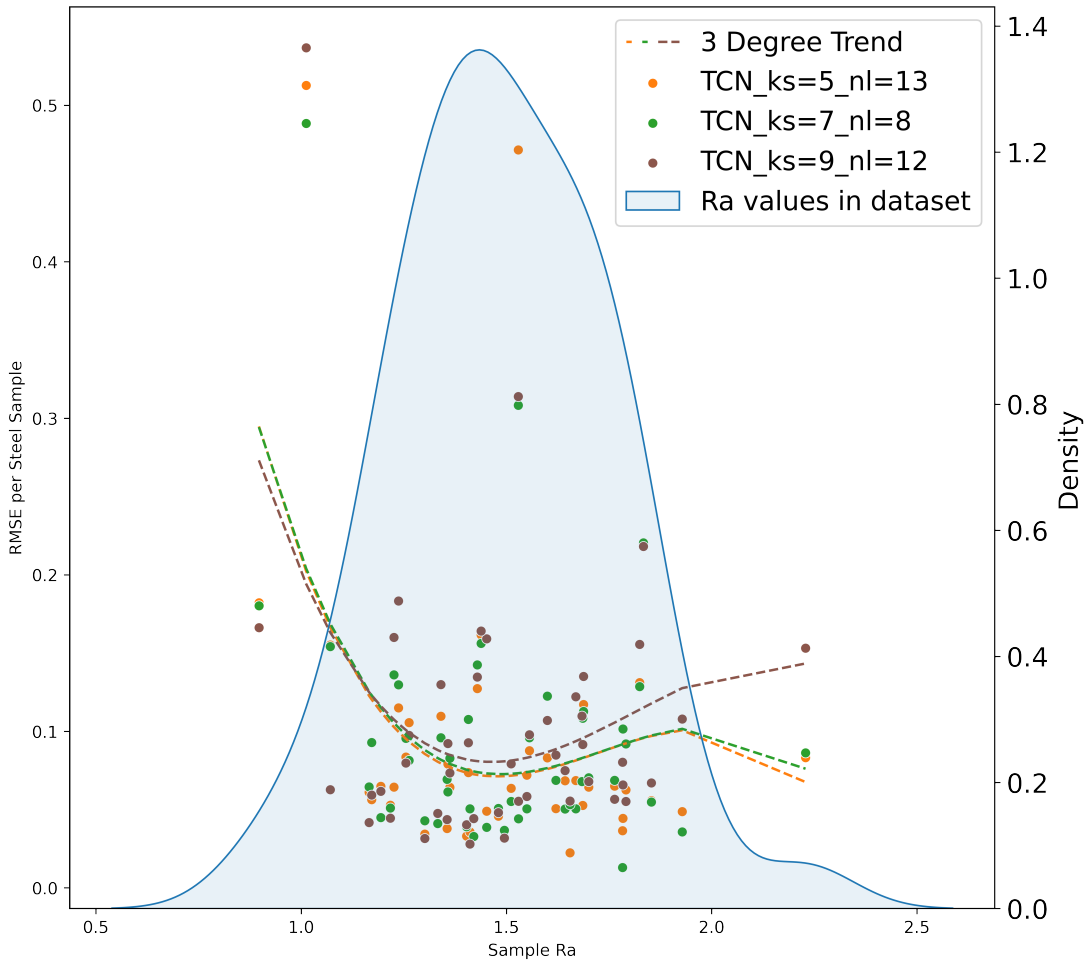


Figure 8.8: The TCN models' results vs the stylus steel sample R_a for each fold. The right axis also shows the density of the R_a values in the dataset.

We hypothesise that the underperformance of transformer approaches can be attributed to the limited size of the available data and the absence of unsupervised pretraining techniques or transfer learning. Typically, transformer networks are trained on significantly larger datasets, such as being pre-trained on extensive corpora and subsequently fine-tuned on the specific experimental dataset. In our study, we did not employ any form of pretraining for the transformer-based approaches, which likely explains their inferior performance compared to the Cov-based models. Cov-based models possess greater inductive bias, enabling them to learn effectively even with less data. On the other hand, transformer models have the potential to achieve superior results by adopting diverse learning strategies but require a larger amount of data to acquire this capability.

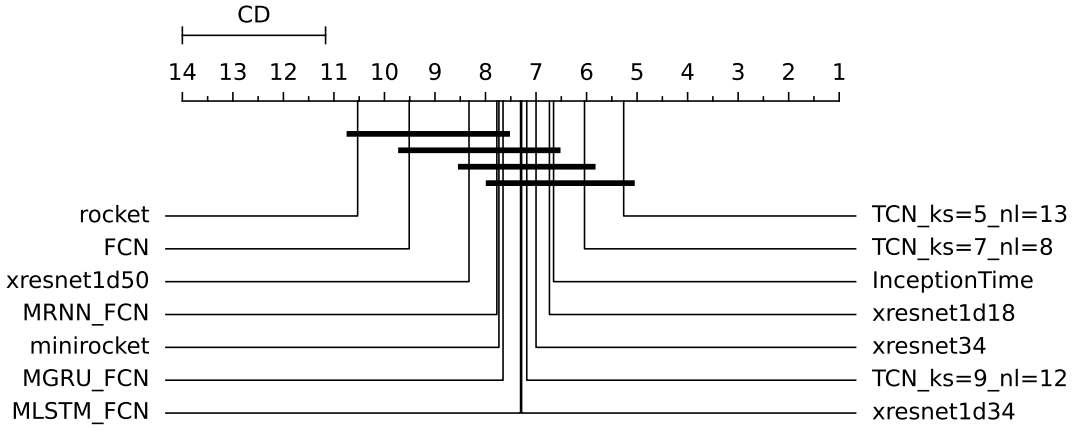


Figure 8.9: Critical difference diagram showing statistical difference comparison of the regression algorithms on the 49 different steel samples.

We conducted a statistical comparison of the algorithms on the k-fold test set results from the 49 steel samples. We follow the work of [113] and adhere to the guidelines outlined by Demšar [30].In this process, each algorithm is ranked for each of the 49 folds according to its Root Mean Square Error (RMSE), assigning Rank 1 to the algorithm with the lowest RMSE and Rank 14 to the highest. In the event of ties, fractional ranking is employed. Subsequently, the average rank for each algorithm is calculated. The Friedman test [41] is then applied to these average ranks. In cases where the null hypothesis was rejected, the post-hoc two-tailed Nemenyi test is employed, following the methodology described by Demšar [30], to compare the algorithms against each other. The performance of the algorithms is deemed significantly different if the average ranks exhibited a difference equal to or greater than the critical difference of 2.8.

Figure 8.9 shows the critical difference diagram. The TCN with ks 5 and nl 13 has the best average rank of 5.27 and is significantly different from ROCKET, FCN, and xresnet1d50. The black horizontal bar shows no significant difference between the algorithms performing better than these three. The computational cost experiments show, however, that MiniROCKET and xresnet34, running on CPU and using 2D convolution respectively are much slower than the competitors.

Table 8.4: Inference Times for Different Models

Model name	Time (s)
xresnet1d18 deeper	2.0213
xresnet1d18	2.4981
TCN nl=8 ks=7	2.6053
xresnet1d18 deep	2.6869
XGBoost	3.4882
FCN	3.6852
TCN nl=13 ks=5	3.8410
ResCNN	4.0047
MRNN-FCN	4.1162
MGRU-FCN	4.1506
MLSTM-FCN	4.2164
xresnet1d34	4.4676
xresnet1d34 deep	4.6736
xresnet1d34 deeper	4.7140
Random Forest	4.9194
ResNet	6.8661
ViT Tiny	8.4807

Model name	Time (s)
XceptionTime	8.6212
TCN nl=12 ks=9	9.1388
InceptionTime	10.8375
xresnet1d50	10.8629
xresnet1d50 deep	11.4460
xresnet1d50 deeper	11.5965
xresnet18 2D	17.9335
ViT Small	18.4366
xresnet1d101	20.6272
xresnet34 2D	32.6909
ConvNeXt Tiny	63.0345
Swin v2 Tiny	95.8745
ConvNeXt Small	107.5767
Rocket (GPU [87])	125.8906
Swin v2 Small	188.7902
MiniROCKET	1632.5267

8.4.3 Computational Cost

We performed a limited study to estimate the relative computational cost of using each of the aforementioned models in production for R_a predictions. In order to do so, we calculated inference times on untrained versions of the models using all 724 samples in our dataset. The results can be seen in Table 8.4. These experiments are performed using an AMD Ryzen 7 2700 Eight-Core Processor and a 2080ti GPU. The reported inference times are a mean average over 10 predictions. For the deep learning experiments, we use a batch size of 8. As expected, neural networks with a larger number of weights take longer to make predictions, with the xresnet1d18 being a smaller model and therefore the fastest to perform. 2D models are also shown to be more computationally expensive than 1D models. MiniROCKET has a significantly larger computational cost than others, due to its lack of implementation for GPU and large number of transformations. In theory, MiniROCKET is a faster variant of the ROCKET model; however we have implemented ROCKET in PyTorch so that it can utilise GPU [87].

These experiments are important to judge the feasibility of using these models with the on-line system at line speed. For our use case, train time is unimportant as the models will be trained off-line. Each data sample is approximately 6.5cm and the line speed is typically

between 50 and 300cm/s, meaning that to perform prediction at line speed we need to be able to run approximately between 8 and 47 examples per second for full coverage. As we have run the experiments with all 724 samples from our dataset, the boundary in the table for a model which can predict at the maximum line speed is 15.4s for all 724 samples. All 1D deep learning models apart from xresnet1d101 are below this threshold. However, this threshold does not rule out feasibility as full coverage is not a hard requirement. Predicting a more accurate R_a less often can be more beneficial for line operators or automatic tempermill control.

8.5 Conclusion

In this chapter, we have provided a comprehensive evaluation of machine learning models for the prediction of the R_a roughness parameter from laser light reflection data.

This research introduces a novel end-to-end methodology that substantially enhances the accuracy of on-line R_a roughness parameter prediction by using machine learning models, an approach previously unexplored. Conventional stylus measurements cannot be taken on-line, and existing on-line methods have been measured to exhibit inaccuracies compared to the industry-standard stylus measurements. Our approach and the models tested consistently outperformed the conventional closed-form baseline, underscoring their effectiveness. This is an alternative approach for enhancing R_a prediction to the one presented in Chapter 6, where we used imputation to improve the surface profiles that are used for the closed-form solution.

Moreover, we validated the robustness of our approach through two distinct experimental setups: the 20% test data experiments and the more challenging k-fold cross-validation. The findings from these experiments highlight not only the potential but also the reliability of our machine learning-based approach, reaffirming its capability to generalise and deliver accurate R_a roughness predictions on unseen samples.

This approach offers fresh insights into how other surface roughness parameters can be predicted and measured. The success of our approach suggests that, in future work, similar models can be employed to predict a spectrum of other steel surface parameters, thereby unlocking a broader range of applications in various domains. By identifying the most effective machine learning models for our specific problem, this work offers a valuable starting point for those encountering similar data challenges.

In the following chapter, Chapter 9, we explore next whether the performance of machine learning when predicting the surface roughness parameter, R_a, can be further improved with the introduction of unsupervised pretraining.

Chapter 9

Improving Roughness R_a Prediction by Pretraining on Unlabelled Production Data

In the previous chapter, we established that a direct, end-to-end deep learning model can significantly outperform the traditional closed-form solution for predicting the surface roughness parameter, R_a. While this supervised approach is effective, its performance is fundamentally limited by the size and quality of the available labelled dataset. This presents a common challenge in industrial settings, where vast quantities of unlabelled sensor data can be collected with ease, but the acquisition of corresponding ground-truth labels is often difficult, expensive, or infeasible.

This chapter investigates whether we can overcome this labelled-data limitation by leveraging the large, unlabelled dataset collected directly from the production line. The central hypothesis is that by pretraining a model on this unlabelled data, it can learn a rich and robust representation of the underlying steel surface characteristics. This learned representation can then be transferred and fine-tuned on the small, labelled dataset, resulting in a model with superior accuracy and generalisation compared to a baseline of fully supervised learning from scratch. The content of this chapter is based on work which has been accepted to the Springer conference Advanced Concepts for Intelligent Vision Systems (ACIVS) 2025 [86].

9.1 Context and Motivation

While the supervised approach is effective, its performance is fundamentally limited by the size of the labelled dataset. This is a common issue in manufacturing, where machine learning shows great promise for process improvement, but collecting labelled samples during live production is often challenging or infeasible.

As detailed in our data chapter (Chapter 4), this challenge is particularly acute for our task. Labelled data collection requires a time-consuming offline process in a lab environment, primarily due to the impossibility of physically aligning the contact-based stylus tool with the non-contact laser scanner at the required microscopic scale. Applying a stylus to a moving production strip is not viable as it would damage the product surface. However, since the laser scanning system is permanently installed on the production line, a vast amount of *unlabelled* data is readily and continuously generated, presenting a key opportunity. The weakly related non spatially aligned measurements of our labelled dataset were time-consuming resulting in only 724 laser-based measurements from 49 steel samples. Conversely, collecting unlabelled laser measurements is far more scalable, resulting in 52,388 measurements from 112 steel coils. This disparity presents an opportunity to leverage the larger unlabelled dataset for pretraining.

We now offer a brief recap of the most pertinent features of our unusual datasets, which have already been introduced thoroughly in Chapter 4. As previously established, the data collected from the laser scanning system is unusual in that it is not a traditional image, but exhibits spatial dependencies across both the width (20 sensors) and length (T) dimensions. This introduces the following unique challenges:

- The long and narrow aspect ratio (20 sensors across a varying length T reaching up to 65K) complicates the use of conventional image-based architectures designed for square or moderately rectangular inputs.
- Variability in surface conditions and manufacturing processes results in diverse reflection patterns, requiring robust feature extraction methods to generalise effectively.
- Limited labelled data hinders the effectiveness of supervised approaches, highlighting the need for pretraining techniques like SSL.
- The misalignment between stylus and laser measurements in the labelled dataset makes analysis difficult.

Table 9.1 summarises the key statistics of the labelled and unlabelled datasets. The labelled dataset captures a limited but diverse set of samples, while the unlabelled dataset provides extensive coverage of the steel surface measurements. These datasets provide a robust foundation for evaluating the effectiveness of representation learning techniques in capturing the unique characteristics of steel surface measurements while fine-tuning on high-quality labelled measurements for accurate surface roughness prediction.

Table 9.1: A summary of the key statistics for the labelled and unlabelled datasets.

Property	Labelled Dataset	Unlabelled Dataset
Data Source	Lab Samples	In-Production
Number of Steel Samples/Coils	49	112
Total Number of Laser Scans	724	52,388
Surface Roughness (R_a) Labels	Available (Mean per sample)	Not Available
Coil/Sample ID	Available	Available
Sequence Length (T)	65,536	50,000
Sampling Interval	$0.8\,\mu$m	$1.0\,\mu$m
Physical Scan Length	≈ 5.24 cm	≈ 5.00 cm

Machine learning has shown great promise for improving key industrial processes; however, progress is often hampered by the difficulty of collecting labelled samples in a production environment. To address this, representation learning and self-supervised learning (SSL) have emerged as powerful approaches for pretraining models in domains where acquiring labelled data is costly, time-consuming, or infeasible.

These methods have been extensively studied in computer vision [19, 23, 46, 50], their success extended to time-series analysis [42, 81, 127, 133], and they have been applied to areas such as manufacturing and materials science [3, 98, 100, 118]. However, their application to non-image spatial industrial data arrays, for surface roughness regression in steel production, remains unexplored. The primary contribution of this chapter is therefore a systematic comparison of pretraining approaches, investigating whether leveraging unlabelled data can significantly enhance predictive performance for this real-world manufacturing problem.

Two distinct perspectives guide our experiments. First, vision-based models treat T as the width (W) and the 20 sensor channels as the height (H), rendering the data analogous to a $20 \times T$ image. However, vision architectures are traditionally optimised for square or moderately rectangular inputs, which can hinder their ability to handle long and narrow data efficiently. By contrast, time-series models treat the 20 sensor channels as independent input features (C)

and T as the temporal axis, leveraging their capacity to capture long-range dependencies. This dual perspective allows us to explore the trade-offs between the efficacy of spatial correlations captured by vision-based models against the sequential dependencies modelled by time-series architectures. Specifically, we compare:

- Baseline supervised models trained only on the labelled dataset without pretraining.
- Models pretrained with contrastive SSL and fine-tuned on the labelled dataset.
- Models pretrained as an autoencoder and fine-tuned for regression.
- Models trained with coil classification as a pretext task, followed by fine-tuning on the R_a prediction task.

Both the contrastive SSL approach and coil classification are specifically tailored for our application. For the former, our framework uses ideas from established contrastive learning methods, such as SimCLR [19] and MoCo [23,50]. However, due to our unique long, thin, and structured data, we require novel customisations and new augmentations. For the latter, we use supervised pretraining with coil classification as a pretext task, predicting the specific coil from which a laser measurement originated. Since each coil contains multiple measurements, this task enables the model to learn coil-specific variations that can improve downstream regression performance. We designed this approach specifically for our unique problem. In addition to both of these, we explore autoencoding techniques using temporal convolutional networks (TCNs) [73] to extract compact representations from this structured data. These techniques are further explained in the next section, where we introduce our methodology.

9.2 Methodology

This section describes the methodology used to investigate representation learning approaches. We explore three main representation learning techniques for the pretraining: (1) contrastive learning for identifying views from the same samples, (2) autoencoding to recreate the input data after being passed through a bottleneck that compresses the latent embedding, and (3) coil classification as a pretraining task. We chose three contrasting approaches in order to leverage different patterns that might help with model training, including those unique to specific coils, and to identify which was the most effective.

In particular, we chose these approaches for the following reasons. Using contrastive learning, we hypothesised that a representation might be learnt that would distinguish between different coils. Recall that our data is comprised of various steel samples, many of which originate from the same coils. Learning to distinguish between these could allow a model to make R_a predictions more representative of the specific coil, as the steel manufacturer intends that each coil will have similar surfaces and properties along its length. As a direct comparison, we included a specific coil classification task, to compare how this performed when deliberately encoded in the pretraining. Finally, the autoencoder represents a different family of approaches which are generative in nature. Generative approaches are well-established in pretraining, so it was imperative to evaluate how these performed on our data. In all cases, the representation learning was followed by fine-tuning on the R_a regression task.

All three approaches, as well as the baseline, use the same TCN model backbone for learning the representation. Here the "backbone" refers to the convolutional layers alone, without the final linear layers used for the regression prediction. During fine-tuning, the pretrained backbone is transferred and adapted for regression by replacing the pretraining-specific prediction head with a randomly initialised linear layer. This transfer learning approach ensures that all methods leverage the same feature extractor model while isolating the effects of the learned representations rather than the regression layer's capacity.

TCN was chosen over other models as it was among the top-performing architectures in the experiments of the previous chapter, while remaining computationally efficient. Rather than selecting the largest or absolute best-performing configuration, we deliberately chose the ks=7, nl=8 variant as a strong but lightweight backbone. This allows us to perform extensive pretraining and fine-tuning experiments, including repeated baseline runs and checkpoint evaluations every 50 epochs, without prohibitive computational cost. The objective of this chapter is therefore not to surpass the strongest architecture overall, but to evaluate how pretraining impacts a fixed and efficient backbone. We use parameters to create a TCN with 8 layers, each with 25 channels and convolutional kernels of size 7. The implementation used is from the tsai Python package [93].

9.2.1 Contrastive Learning Approach

Contrastive learning aims to learn representations by bringing augmented views of the same sample closer in the feature space while pushing apart representations of different samples.

Our data has unique characteristics in length, width, and nature that require significant

adaptations compared to other contrastive learning pipelines. Typically in the image domain, the positive samples are generated by applying two random croppings on the same image. Each crop will then have augmentations applied to further improve generalization and diversity of the input data. However, this approach does not make sense for our data as all data will always have all 20 sensors; therefore, learning representations without this aspect might have counterproductive impacts. Therefore, we split each $20 \times T$ sample along the length dimension into two non-overlapping views. These views retain the full 20 sensor channels while sampling less than half of the length T, with a gap of 10 in the T dim between them to prevent edge artifacts from being used to match positive pairs.

Custom augmentations are specifically tailored to enhance the dataset for contrastive learning. First, brightness and contrast variations are introduced through a widely used colour jitter transformation, a standard augmentation in contrastive learning pipelines such as SimCLR. This transformation modifies intensity values to simulate varying sensor conditions, with each application being probabilistic. Since our data consists of grayscale images, only the brightness and contrast components of the transformation are applied, while hue and saturation adjustments are inherently excluded as they do not apply to grayscale data. Next, Gaussian blur is applied with kernel sizes of 5 and sigma values between [0.1, 1.3] to mimic signal diffusion and variability in the data. Bright channel artifacts, where a single sensor exhibits consistently high intensity across all time steps, are mitigated by applying an augmentation that assigns a randomly selected sensor row a high value based on the top percentile of the data. Spatial augmentations follow, with random horizontal and vertical flips applied independently with a 50% probability each, introducing more spatial variability to account for potential symmetries. Noise is then introduced by adding Gaussian noise or row-specific noise to individual sensor readings, adding more data variability, simulating real-world sensor variation. Signal strength is modulated through a Weakening Threshold transformation, which replaces lower-intensity values in the input tensor with their mean. This is achieved by first randomly selecting from the tensor a low threshold, derived from the minimum value and a random factor. All values below this threshold are replaced with the mean of the removed values, ensuring that the transformation selectively weakens smaller intensities while maintaining the integrity of higher-intensity regions. This is applied at random, increasing the diversity of the dataset. Intensity shifts are introduced that adjust each sensor row's intensity independently, based on the overall intensity of the other rows. For each sensor row, there is a probabilistic chance of applying a shift. The shift value is calculated based on the mean intensity across all sensor rows, scaled

by a user-defined maximum shift factor and further modified by a random multiplier to introduce variability in both magnitude and direction. By subtly altering the intensity values of sensor rows, this transformation encourages the model to focus on learning representations that are invariant to minor brightness differences among the 20 sensors, promoting robustness to sensor-specific artifacts while preserving the spatial and temporal correlations of the data. Finally, a low-intensity Gaussian noise is injected to further simulate fine-grained real-world noise while maintaining the data's overall integrity.

Each of these augmentations is applied with a 50% probability to maintain diversity within the dataset while avoiding excessive alteration. The augmentations were selected to address specific deficiencies in the dataset, such as noise in some samples, irregular thresholding, and sensor rows that tend to exhibit consistently high or low intensities. By applying these transformations randomly to both views of a sample, the model is compelled to match representations despite the variations introduced. This forces the model to learn features that are invariant to these artifacts and adapt to the inherent variability of the dataset. Collectively, these augmentations enable the model to capture robust and generalized representations, ensuring it effectively learns the underlying patterns in the data.

Contrastive Learning performs better when large numbers of negative samples are in each batch. Due to the large size of the dataset and the high memory demands of the model, the batch size is limited to 12 samples to fit on the GPU memory. To address this, we employ the queue-based memory mechanism from the MoCo framework [23, 50], which maintains a dynamic queue of negative samples from previous batches.

9.2.2 Autoencoding

Our autoencoding model is a novel contribution of this thesis, whereby a TCN architecture is used for both the encoder and a decoder [1]. Our custom-designed TCN-based autoencoder is employed to learn compact representations of laser reflection data by reconstructing the input from a compressed latent embedding. The architecture consists of a sequential pipeline comprising an encoder, bottleneck, and decoder, designed to process the unique characteristics of the dataset efficiently.

The encoder uses the backbone from the TCN and learns a representation that is then compressed using a bottleneck. The bottleneck is implemented using a custom module, which

[1]The Python code for this TCN autoencoder model is available at: `https://github.com/alexander-milne/tcn_autoencoder`

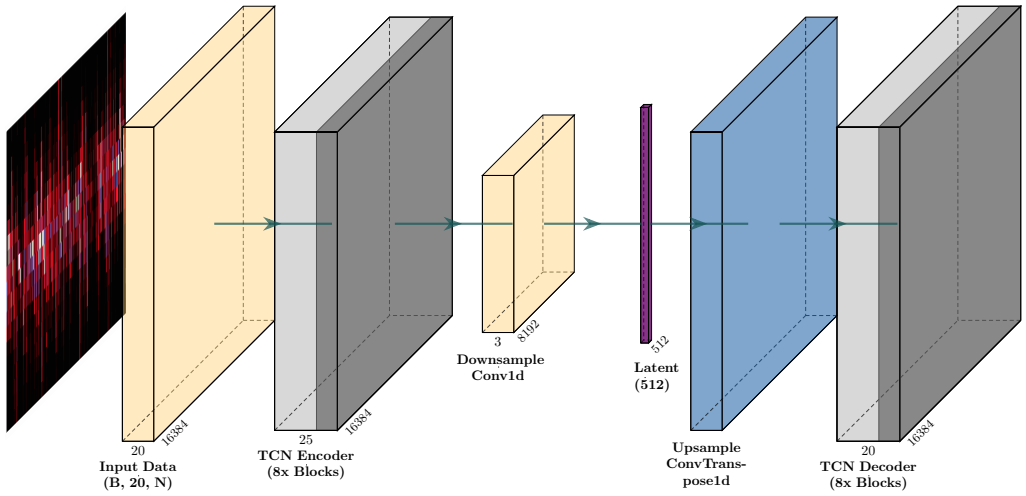


Figure 9.1: The overall TCN-based autoencoder architecture, showing the main stages: a TCN encoder, a bottleneck for dimensionality reduction, and a symmetric TCN decoder for reconstruction.

employs a convolutional layer with a stride of 2, followed by a flattening operation and a fully connected layer that compresses the data into a latent embedding of size 512, with 3 bottleneck channels.

The bottleneck is then reversed, expanding the latent embedding back into a higher-dimensional space. The decoder, which is an inverse of the encoder, then reconstructs the original input data from the expanded representation, ensuring the recovery of meaningful patterns within the laser reflection data. These choices strike a balance between dimensionality reduction and retaining critical features for reconstruction.

9.2.3 Coil Classification

In the coil classification pretraining task, the model predicts the specific steel coil from which a laser sample was taken. This supervised pretraining task is designed to leverage the inherent similarities within the same coil and variability between other coils as a source of meaningful features for downstream regression. The dataset includes coil IDs for both labelled and unlabelled laser measurements. The pretraining process is applied to the entire unlabelled dataset and the train split from the labelled data. We use a TCN model for this task, with coil ID as the prediction label.

9.2.4 Evaluation Framework

To evaluate the effectiveness of pretraining, we compare our fine-tuned model against a baseline model trained solely on the labelled dataset without any pretraining (random initialisation). Before fine-tuning a pretrained model, we replace the head with a randomly initialised regression head. Baseline and pretrained models use the same architecture and are trained under identical conditions, allowing for a direct comparison of the benefits of starting from a pretrained state, leveraging unlabelled data. Therefore, we analyse the effectiveness of each representation learning approach in improving downstream regression performance against one another and the baseline.

We assess the performance of all approaches using several key metrics: root mean squared error (RMSE), correlation, maximum absolute error (the largest deviation from the ground truth), and prediction coverage. Prediction coverage represents the percentage of samples where predictions fall within the range defined by the minimum and maximum R_a measurements, reflecting the natural variability of surface properties at different spatial locations as captured by multiple stylus-based measurements within a steel sample.

Experiments were conducted across two machines, one equipped with a single NVIDIA RTX 4090 GPU and the other equipped with multiple NVIDIA RTX 3090 GPUs. Pretraining was performed on the 3090 machine and fine-tuning on the 4090 machine. These details are provided for completeness; hardware differences are not expected to materially affect the reported performance metrics. Due to the large sample size and high memory requirements, batch sizes were kept small (12 for pretraining and 4 for fine-tuning), and the MoCo framework was used for contrastive learning to handle the small batch size effectively.

9.3 Results

In this section, we compare the performance of our representation learning approaches on the R_a surface roughness prediction task to the use of a fully supervised baseline trained from scratch on the labelled dataset. To evaluate the efficacy of each method, we recorded the fine-tuning performance of models pretrained for various epochs, saving weights at intervals of 50 epochs. This analysis allows us to examine how the progression of pretraining impacts fine-tuning results on the R_a regression task.

9.3.1 Fine-Tuning Performance

Table 9.2 summarises the performance of all models, including a baseline supervised model trained from scratch. The baseline results were obtained by training the model on the fine-tuning data without any pretraining.

Across the course of experimentation, the baseline (trained from scratch) was run 292 times. Initially, a number of standalone baseline experiments were performed. In addition, every fine-tuning experiment begins by training from scratch before loading pretrained weights, and these baseline stages were also recorded. Many of these fine-tuning runs were exploratory or terminated early; however, their from-scratch phases were still valid and were therefore included in the statistical analysis. Consequently, the total of 292 runs reflects the cumulative number of baseline executions collected throughout development. We average the metrics over these runs, with the standard deviation provided to account for variability.

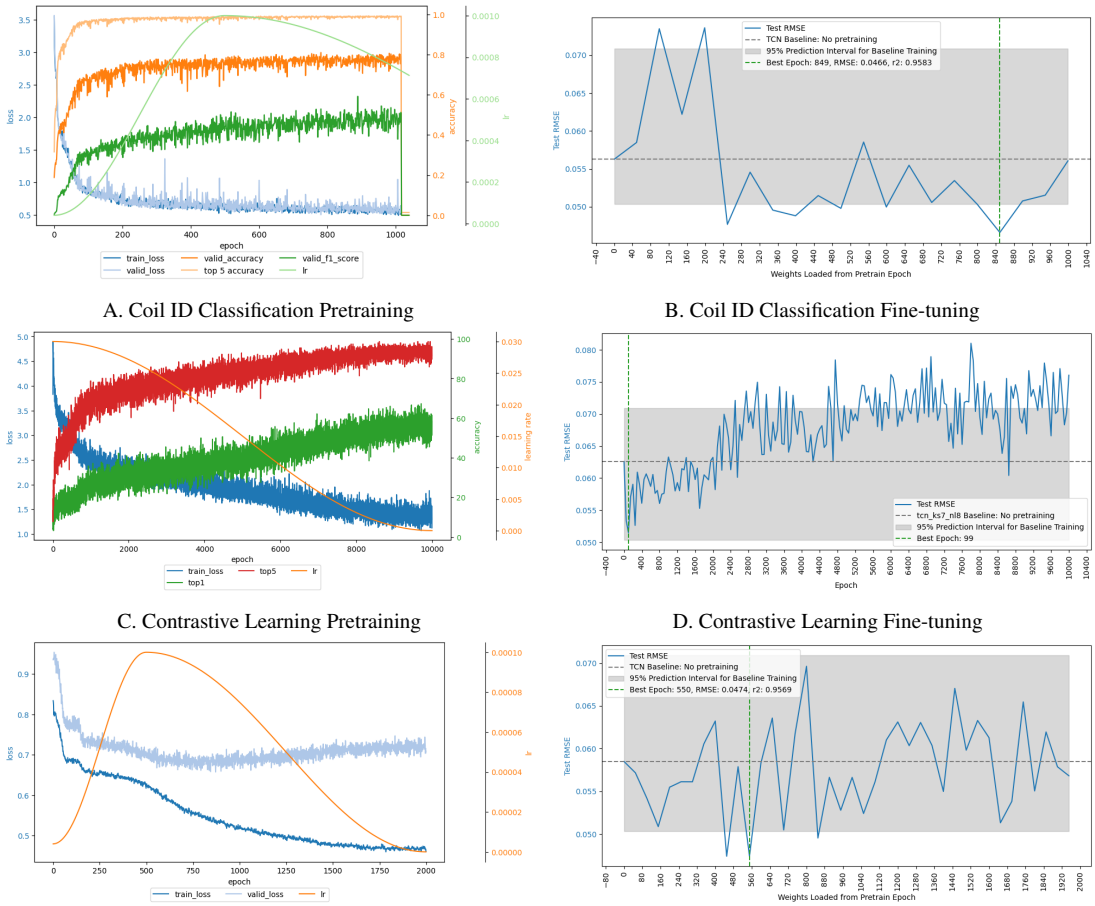
For models using pretraining, we report results based on the best fine-tuning performance achieved when fine-tuning from pretrained weights saved at specific pretraining epochs. Fine-tuning was performed every 50 epochs of pretraining to observe how performance varies with the amount of pretraining.

Due to both computational and time constraints, it is not feasible to run the full pretraining-to-fine-tuning pipeline multiple times for each approach. The 95% percentile and standard deviation from the 292 baseline runs are therefore used to assess whether performance improvements exceed stochastic variation and stem from the quality of the pretrained representations rather than stochastic effects.

Table 9.2: Quantitative Results: Fine-Tuning Performance After Pretraining

Method	**RMSE ↓**	**Correlation ↑**	**Max Abs. Error ↓**	**Coverage (%) ↑**
TCN Baseline (Mean ± Std)	0.0585±0.0061	0.9336±0.0142	0.1937±0.0402	62.0516±4.4103
Contrastive SSL	0.0515	0.9491	0.1529	64.4737
Coil Classification	**0.0466**	**0.9583**	0.1598	**69.7368**
TCN Autoencoder	0.0474	0.9569	**0.1381**	66.4474

Results table 9.2 demonstrates that all of the pretraining methods outperform the baseline



A. Coil ID Classification Pretraining

B. Coil ID Classification Fine-tuning

C. Contrastive Learning Pretraining

D. Contrastive Learning Fine-tuning

E. Autoencoder Pretraining

F. Autoencoder Fine-tuning

Figure 9.2: These plots show a comparison of pretraining metrics and the resulting best fine-tune loss when fine-tuning is performed with starting weights from the corresponding pretrained epoch. On the left are the pretraining loss curves, while on the right are the curves for the fine-tune loss. Fine-tuning results show the RMSE on the y-axis; therefore, lower is better. Results are better when these values show a consistent trend lower than the 95% gray box such as in 9.2B. In the legend, "lr" denotes the learning rate used during fine-tuning.

when compared to the mean value of the four metrics. The backbone architecture used in this chapter corresponds to the TCN ks=7, nl=8 configuration evaluated in Chapter 8; therefore, the relevant comparison is between the pretrained and non-pretrained versions of this same model. For this fixed backbone, coil ID classification pretraining reduces the RMSE from 0.0585 ± 0.0061 to 0.0466, representing a clear and statistically meaningful improvement beyond the baseline variability. To further contextualise this improvement, we compare the performance of these approaches to the 95% percentile in Figure 9.2. The coil classification pretraining task shows the best performance for most of the metrics, outperforming all other approaches

on RMSE, correlation and coverage, while the TCN autoencoder performs better on maximum absolute error. While maximum absolute error gives some indication of overall performance, the poor performance of the coil classification approach on this metric may be due to a single outlier. Given more availability of time and compute, it would be more beneficial to rerun these approaches and calculate a mean absolute error instead. Overall, on these metrics coil classification performs best of the four methods.

While larger models evaluated in Chapter 8, such as xresnet34, achieve slightly lower absolute RMSE values, they incur substantially higher computational cost. The objective of this chapter is not to surpass the strongest architecture overall, but to evaluate how pretraining improves a fixed and computationally efficient backbone. Within this controlled setting, pretraining demonstrably enhances performance.

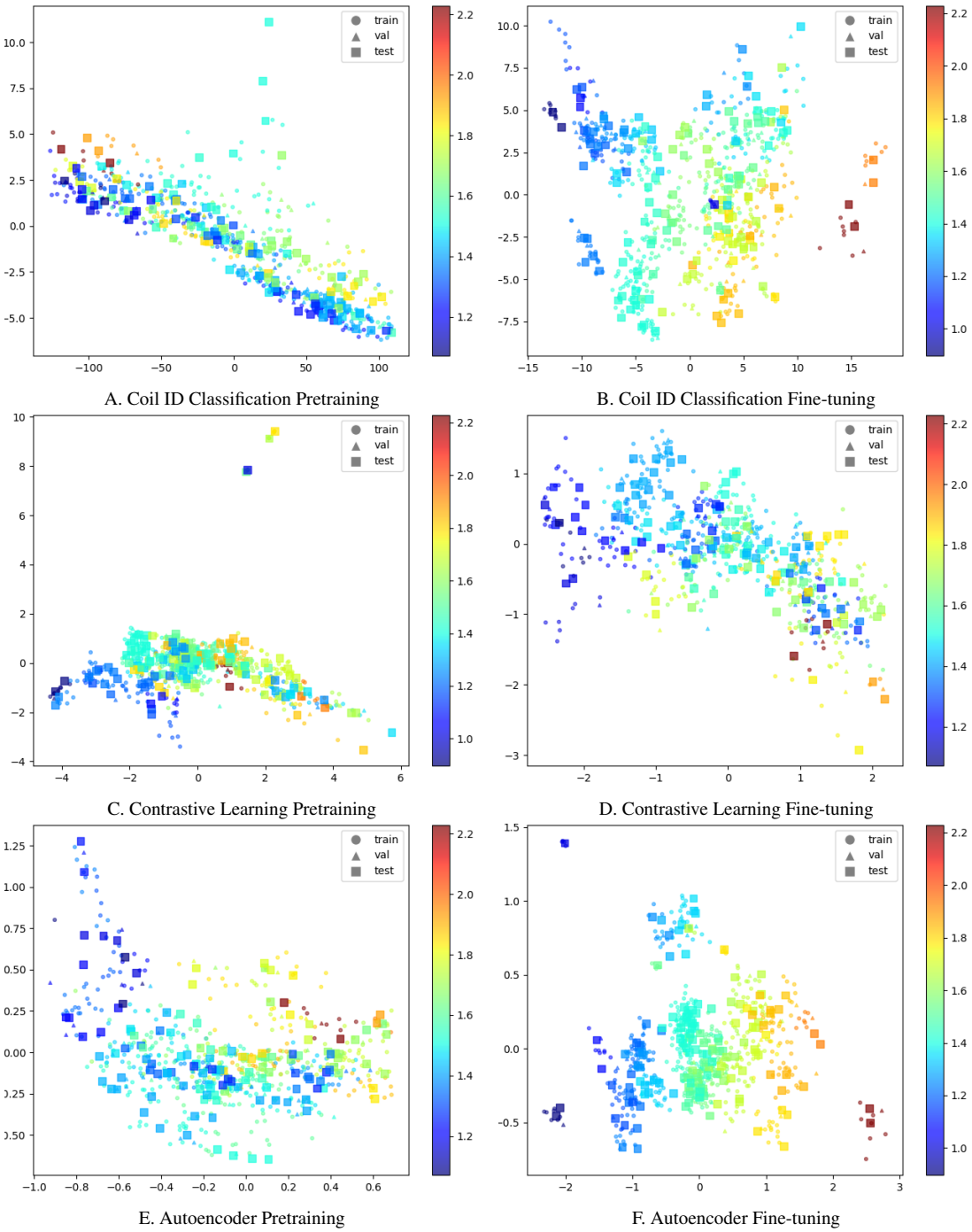
Figure 9.2A. demonstrates how fine-tuning performance varies when the model is pretrained for a varying number of epochs using the coil ID classification pretraining task. Initially, the pretrained model performs worse than the TCN baseline model, but at 250 epochs of pretraining, a large improvement is observed. From here, pretraining continues to show an improvement over the baseline, with the best performance observed after 850 epochs of pretraining for this specific training iteration. The performance of the pretrained model is an improvement over the 95% threshold at several points along the pretraining curve.

By contrast, other methods show no improvement over the baseline model trained only on the fine-tuning data. In figure 9.2D., we observe that the contrastive learning approach has a negligible impact on performance up to epoch 2000, performing similarly to the model which received no pretraining. Beyond this point, additional pretraining leads to negative learning, resulting in worse fine-tuning outcomes relative to the baseline.

The TCN autoencoder does not demonstrate a consistent improvement over the baseline. As shown in Figure 9.2F., the fine-tuning performance exhibits high variability across pretraining epochs, with no clear sustained trend relative to the baseline threshold. While certain checkpoints achieve large improvements over the baseline, others perform substantially worse, indicating that this pretraining objective introduces significant variability in downstream performance without a consistent directional benefit.

9.3.2 Discussion and Qualitative Analysis

To further assess the impact of different pretraining strategies on the main regression task of predicting Ra from laser surface reflection data, we analyze the learned representations



A. Coil ID Classification Pretraining

B. Coil ID Classification Fine-tuning

C. Contrastive Learning Pretraining

D. Contrastive Learning Fine-tuning

E. Autoencoder Pretraining

F. Autoencoder Fine-tuning

Figure 9.3: These plots display the first two principal components from PCA. The R_a values are represented by color, illustrating the representation spaces learned during pretraining and fine-tuning across the different methods.

using Principal Component Analysis (PCA). Specifically, we visualize the first two principal components of the representations learned through each pretraining method, both before and after fine-tuning. The points in these plots are colored according to their corresponding Ra values, forming a gradient that ideally reflects a well-structured representation space. A high-quality representation should exhibit a smooth and well-ordered gradient, indicating that the learned features effectively encode information relevant to Ra prediction.

9.3.3 Pretraining Representations

The PCA visualization of representations learned through contrastive pretraining does not exhibit a well-structured Ra gradient. The colors appear largely disorganized, with significant overlapping of different Ra values. Despite this, there is a mild gradient present, suggesting that some degree of useful structure has been captured. However, the lack of clear ordering and separation of colors indicates that contrastive learning, in this setting, does not effectively align the representation space with the regression task.

The autoencoder pretraining yields representations that are exceptionally well-structured. The PCA plot shows a strong and continuous gradient of Ra values, with clear separation and correct ordering of points. This suggests that the autoencoder effectively captures meaningful features relevant to the regression task, producing a representation space that is inherently aligned with the target variable even before fine-tuning.

The Coil ID classification pretraining results in a representation space that is more structured than that of contrastive learning, though far from as clean as the autoencoder. While a gradient is present, there are regions where points of different colors overlap, indicating mild misalignment. This suggests that while the Coil ID classification pretext task captures some useful features, it may not fully align with the requirements of the regression task, leading to minor inconsistencies in representation quality.

Notably, these results highlight an important consideration: the effectiveness of a pretraining task in structuring representations is not guaranteed when evaluated against a downstream task that differs from the original pretext objective. A misaligned pretraining task may still yield representations useful for the pretext objective but suboptimal for the target task.

9.3.4 Fine-tuned Representations

Following fine-tuning, we expect to observe improved structure in the representation space, as the network learns to adapt its features specifically for the Ra regression task. The PCA

visualizations confirm this expectation, though with notable differences across methods.

The contrastive learning representations exhibit mild improvements after fine-tuning but still retain some degree of disorder. Overlapping points and misalignment persist, with instances of high-Ra values (e.g., red points) appearing in regions that should correspond to lower Ra values. Additionally, many blue points, indicative of low Ra values, are misplaced within clusters of higher values. These inconsistencies suggest that while fine-tuning helps, the initial suboptimal structure of contrastive pretraining representations limits the final performance.

The autoencoder-based representations remain highly structured after fine-tuning, maintaining a near-perfect gradient. While a few minor overlaps exist, particularly between light green and light blue points, the overall structure remains highly coherent. The clear and continuous gradient suggests that autoencoder pretraining provides a highly favourable initialisation for fine-tuning, allowing the network to retain and refine a meaningful representation space for Ra prediction.

The Coil ID classification pretraining also leads to well-structured fine-tuned representations, with a strong gradient and minimal overlap. However, a few dark blue points appear misplaced within mid-range Ra clusters, which may correspond to the larger maximum absolute error observed in the quantitative results. Despite this minor discrepancy, the overall structure remains strong, indicating that Coil ID pretraining effectively aids in learning a meaningful feature space for the downstream regression task.

These qualitative results reinforce the findings from our quantitative analysis. While all pretraining strategies benefit from fine-tuning, the autoencoder pretraining consistently produces the most well-structured representations, both before and after fine-tuning. The Coil ID classification pretraining also performs well, albeit with minor inconsistencies. In contrast, contrastive pretraining struggles to produce well-ordered representations, potentially due to a misalignment between the pretext task and the regression objective. These findings highlight the importance of selecting a pretraining approach that aligns well with the downstream task, ensuring that learned representations are not only transferable but also structurally meaningful for fine-tuning.

Both autoencoder and Coil ID pretraining demonstrate clear advantages, as they lead to more structured representations that align well with the Ra regression task. Contrastive pretraining, on the other hand, shows weaker performance in this regard, suggesting that its learned representations do not provide as beneficial an initialization for fine-tuning. Contrary to our

hypothesis, this pretraining approach caused a degradation in fine-tuning loss rather than an improvement. These findings emphasize the importance of selecting a pretraining method that effectively transfers knowledge to the downstream task, ultimately leading to improved fine-tuning performance over the baseline.

9.4 Conclusion

In this chapter, we have explored the application of representation learning techniques to improve predictions of surface roughness (R_a) from structured non-image data in steel production. To isolate the effect of pretraining, we fixed a computationally efficient backbone (TCN ks=7, nl=8) rather than selecting the absolute strongest architecture from Chapter 8. Within this controlled setting, we demonstrate that pretraining can meaningfully enhance downstream regression performance.

Specifically, coil ID classification pretraining showed significant improvements over the non-pretrained TCN baseline (RMSE 0.0585 to 0.0466), yielding the best results across multiple metrics for this backbone architecture.

However, variability in pretraining outcomes, particularly with the contrastive learning and autoencoder approaches, has highlighted challenges associated with aligning pretraining objectives to downstream tasks. The observed negative learning effects in some cases underscore the importance of task-specific pretraining strategies that effectively capture the unique characteristics of structured industrial data.

Overall, we have shown that pretraining can leverage large unlabelled datasets to mitigate the constraints of limited labelled data, providing a meaningful initialisation for fine-tuning. Further research is required to refine pretraining techniques, investigate methods to reduce variability across runs, and explore hybrid approaches that combine pretraining objectives more directly aligned with regression tasks. These directions offer promising pathways to improve representation learning for industrial applications and expand its potential beyond traditional image-based data.

Though we have touched on directions for future research, we will contextualise these with future work from other chapters in greater detail in the next chapter, Chapter 10. The next chapter will also conclude the thesis with closing remarks, and a summary of our contributions.

Chapter 10

Conclusion and Future Directions

In this thesis, we have explored the application of machine learning techniques to solve problems in the steel manufacturing industry, specifically by designing and evaluating approaches to improve on-line roughness estimation. We began by characterising potential data loss in the roughness profiles gathered by our industrial partner, and investigating approaches to reconstruct these gaps using machine learning. This data was essential for the fulfilment of our main goal: to predict the roughness of steel samples from the profiles. Before tackling this main goal, we also performed experiments for pruning and fine-tuning of the ROCKET model, as this technique was well-suited for our main experiments. To further improve the efficiency, we also presented a GPU implementation.

These experiments provided a foundation on which we could address our central objective, the prediction of the R_a roughness parameter. In pursuit of this goal, we investigated the efficacy of a wide variety of models including 1D models designed for time series as well as 2D models adapted from computer vision for our task. We then further expanded on this work by incorporating different approaches for pretraining to understand how they affected the final performance. Our contributions are summarised in the following list, before we expand with further details on the work of the individual contributions:

- The development of a methodology for identifying poor-quality data segments in laser reflection measurements, drawing on a combination of three quality metrics
- A study evaluating the performance of contrasting types of machine learning model for the imputation problem arising from missing data, including the incorporation of a novel statistics-based loss

- An efficient GPU-accelerated implementation of the previously proposed ROCKET model
- An experimental evaluation of fine-tuning approaches on a pruned set of kernels to assess the impact on ROCKET's accuracy and efficiency
- A thorough comparative study of various types of machine learning approach to predict the R_a roughness parameter of steel surfaces directly from high aspect ratio input data, with implications for other similarly high aspect ratio datasets
- The evaluation of multiple pretraining strategies that leverage unlabelled data for better representation learning
- Included in this evaluation is a novel autoencoder architecture which incorporates a TCN as both the encoder and decoder
- Analysis of experimental results to identify several machine learning models that demonstrate improvements over baseline approaches, and optimal strategies for predictive accuracy in this domain problem

10.1 Review of Research Questions and Hypotheses

In this section, we now revisit the research questions and hypotheses detailed in Chapter 5 in light of the work presented in the subsequent chapters.

In Chapter 6, we developed our methodology to address data gaps and quality issues within the on-line laser reflection measurements of our dataset. This was one avenue for addressing our main goal of predicting surface roughness, as poor data quality compromises the accuracy of R_a calculations. We posed the following research question:

> *R1: How can missing data gaps in steel surface profiles be effectively imputed such that reconstructed data accurately preserves key roughness statistics and supports reliable surface characterisation?*

In order to address this question, we first designed a novel, multi-metric framework to automatically identify and characterise these poor-quality segments in the industrial sensor data. We assessed different metrics and finally incorporated three scores into our approach: the log-normalised intensity score, the global variation score, and the focus percentage score. To

rigorously evaluate solutions, we then formulated a controlled problem by introducing statistically similar, artificial gaps into high-fidelity stylus profilometry data, which served as a known ground truth. We compared state-of-the-art deep learning models against benchmark classical interpolation methods for the task of imputing these gaps. Our experiments revealed a critical trade-off: models trained with a standard loss function excelled at pointwise accuracy but failed to preserve the crucial statistical texture that defines R_a. To resolve this, we introduced a statistics-aware hybrid loss function that directly optimises for the preservation of R_a. We concluded from our experiments that an attention-based model (SAITS) trained with this custom loss function was able to significantly outperform all baselines, simultaneously achieving high pointwise accuracy and superior statistical fidelity. Thus, we verified our first hypothesis:

> **H1:** A deep learning model trained with a **custom, statistics-aware loss function** that directly penalises deviations in R_a will outperform both traditional interpolation methods and standard deep learning models (trained on MSE alone) in preserving the statistical integrity of the reconstructed surface profile.

Having addressed the problem of data quality, we then investigated methods for improving the efficiency and accuracy of the ROCKET model, as this was a promising approach in the prediction of R_a. Our second research question was as follows:

> *R2: How can ROCKET-based models be optimised for the industrial prediction of steel surface roughness, firstly by improving computational throughput via GPU implementation, and can efficiency improvements allow fine-tuning to improve predictive accuracy with better model efficiency?*

In Chapter 7, we identified that while ROCKET is fast to train, its reliance on 10,000 random kernels creates a significant computational bottleneck during inference, and its non-differentiable nature prevents the kernels from being fine-tuned for a specific dataset. Our contributions to overcome these limitations were presented in two parts. First, we engineered an efficient, GPU-accelerated ROCKET framework and introduced a novel differentiable pooling function, 'softPPV', making end-to-end training possible. Second, we built upon this framework to investigate a new methodology that first prunes the large set of kernels to a much smaller, computationally manageable subset using feature selection, and then fine-tunes the weights of this pruned subset. Though we demonstrated the speed benefit of our efficient implementation of ROCKET, our results implied that fine-tuning the weights of the ROCKET model does not offer a benefit.

This research prepared us for the central task of this thesis: evaluating approaches for predicting the roughness parameter, R_a, directly, in order to identify optimal strategies for doing so. In Chapter 8 we presented our comprehensive study in line with this primary goal, which relates to our third research question:

> *R3: Which machine learning model architectures and techniques provide accurate and robust predictions of the R_a roughness parameter from noisy, high-volume, industrial laser sensor data?*

We began by formulating the problem as a direct, end-to-end machine learning task, training models to learn the mapping from raw laser reflection data directly to the final R_a value. Our main study incorporated a diverse selection of model architectures, including non-deep learning methods, 1D models designed for time series, and 2D models from computer vision. To ensure robustness, we evaluated the models using both a standard hold-out test set and a more challenging leave-one-steel-sample-out cross-validation. We concluded that data-driven approaches consistently and significantly outperform the closed-form baseline. Specifically, a 2D convolutional neural network (xresnet34) achieved the highest overall accuracy, while a 1D Temporal Convolutional Network (TCN) offered the best balance of accuracy and computational efficiency. This study was able to verify our second hypothesis:

> **H2:** At least some ML approaches are able to outperform the closed-form physics-based calculation performed by the SORM machine.

Finally, in Chapter 9, we investigated whether the performance of the supervised models developed in the previous chapter could be improved by incorporating pretraining to address the limitations of our small labelled dataset. This work was in pursuit of our final research question:

> *R4: Can representation learning using large-scale unlabelled production sensor data improve the accuracy, robustness, and generalisation of machine learning models for Ra surface roughness prediction?*

To achieve this, we leveraged a vast, unlabelled dataset collected with from the production line, hypothesising that pretraining on this data would enable a model to learn a rich representation of steel surface characteristics before being fine-tuned on the smaller, labelled set. We designed and compared three distinct pretraining strategies: self-supervised contrastive

learning with custom augmentations; a TCN-based autoencoder; and a supervised pretext task of classifying the unique steel coil ID. We then evaluated the effectiveness of each strategy by fine-tuning the pretrained models on the final R_a regression task. We concluded from our experiments that both the coil classification and autoencoder pretraining strategies led to significant performance gains over the supervised-only baseline, with the coil classification approach yielding the best quantitative results. Finally, we performed an analysis of the learned representation spaces, which confirmed that these successful pretraining methods produced a well-structured feature space that was highly aligned with the final regression task. This result verified our third hypothesis:

> **H3:** Pretraining on large-scale unlabelled production data via representation learning will enhance supervised learning performance for R_a prediction, yielding models with improved accuracy.

Our final hypothesis related to the work of all of Chapters 7- 9:

> **H4:** ML models specifically adapted or optimised for the unique characteristics of steel surface sensor data, such as through tailored pruning, custom loss functions, feature engineering, or architectural innovations, will outperform generic unadapted ML approaches in predicting R_a.

In some cases this was proven to be true, for example in Chapter 9 where the coil classification pretraining task which directly related to our problem setting achieved the best results. Nevertheless, it was not true in all cases. For example, the fine-tuning of the ROCKET model weights did not demonstrate an improvement over a version without fine-tuning.

Through these contributions, we have addressed the key challenge presented by our industrial partner of improving surface roughness prediction with the use of machine learning. Nevertheless, there remain open questions which can be the subject of ongoing research. One significant direction for future research is the adaptation of our presented approaches to other industrial applications, including further challenges in steel manufacturing. In the final section of this thesis, we will summarise some of the open directions for further research in the imminent future and beyond.

10.2 Cross-Chapter Analysis of Methods and Methodology

The progression of this thesis reflects an evolving strategy toward achieving robust, on-line roughness estimation using laser-based measurements in an industrial temper rolling process. The following analyses findings from Chapters 6 through 9 to identify the optimal modelling approaches under varying industrial constraints.

In Chapter 6, we sought to improve the reliability of the existing closed-form laser-based roughness calculation by identifying and imputing poor-quality segments. While the proposed statistics-aware imputation model produced reconstructions that preserved key roughness statistics in controlled experiments, validation was fundamentally limited by the absence of ground truth surface profiles for the laser measurements. This raised deeper questions regarding whether discrepancies between the closed-form solution and stylus profilometry arise solely from missing data, or from intrinsic differences between laser reflection measurements and physical stylus measurements. Additionally, the identification of the poor segments relied on three manual heuristic functions derived from industrial partner insight. It remains an open question whether these functions correctly or optimally isolate erroneous data. Therefore, although imputation improves local reconstruction fidelity, it does not fully resolve this underlying modality gap.

Chapter 7 therefore shifted focus from improving the closed-form calculation to directly predicting R_a using data-driven models. This problem reframing was necessitated by the fundamental absence of spatially aligned ground truth surface profiles for the laser measurements. By targeting the mean R_a value directly, we bypassed the issue with corresponding surface profiles, treating the task as a weakly supervised learning problem where available labels represented the aggregate mean of the steel sample rather than the precise local R_a of a surface profile derived from each individual laser segment. The ROCKET family produced promising predictive results, and we demonstrated that our GPU-accelerated implementation substantially improves throughput, particularly as input dimensionality and dataset size increase. However, attempts to further enhance predictive performance through kernel pruning and fine-tuning did not yield consistent gains.

Chapter 8 expanded our investigation into a comprehensive comparative study across diverse architectural families, including classical models, 1D time-series networks, and 2D convolutional architectures adapted from computer vision. These experiments demonstrated that deep convolutional approaches significantly outperform the closed-form baseline. The xresnet34 architecture achieved the highest absolute predictive accuracy. Within the 1D temporal

convolutional network (TCN) family, larger variants (e.g., ks=9, nl=12) marginally outperformed the more compact configurations. However, we selected the TCN ks=7, nl=8 configuration as the optimal candidate for subsequent representation learning experiments and industrial deployment. Although strictly less accurate than both the xresnet34 and larger TCNs, it offered statistically similar performance at a substantially lower computational cost, maximising efficiency without compromising practical utility.

Finally, Chapter 9 investigated whether leveraging large-scale unlabelled production data could further improve the robustness and accuracy of this selected architecture. Coil classification pretraining reduced the RMSE of the TCN ks=7, nl=8 backbone from 0.0585 to 0.0466, demonstrating that representation learning can meaningfully enhance a computationally efficient architecture without increasing model complexity.

Taken together, these chapters illustrate a clear trajectory: from refining the closed-form solution, to adopting direct data-driven prediction, to systematically identifying high-performing architectures, and finally to enhancing lightweight models through representation learning. While xresnet34 achieves the strongest absolute predictive performance, the pretrained TCN ks=7, nl=8 configuration offers the most favourable balance of accuracy, scalability, and computational efficiency for practical on-line deployment.

Summary of Comparative Findings

Across the thesis, the strongest methods under different criteria can be summarised as follows:

- **Top three predictive accuracy overall:**
  1. **xresnet34** (Chapter 8): 0.0453 RMSE (Best raw accuracy, training from scratch).
  2. **Pretrained TCN ks=7, nl=8** (Chapter 9): 0.0466 RMSE (Enhanced via coil classification pretraining).
  3. **TCN ks=9, nl=12** (Chapter 8): 0.0487 RMSE (Best performing TCN variant trained from scratch).
- **Best method for statistically faithful gap reconstruction:** SAITS with statistics-aware loss (Chapter 6)

For industrial deployment where inference latency and scalability are critical, the pretrained TCN provides the most practical solution, offering strong predictive accuracy while

maintaining computational efficiency. However, if maximum predictive performance is prioritised and sufficient computational resources are available, 2D convolutional architectures such as xresnet34 represent the strongest overall modelling approach.

These findings indicate that the optimal solution depends on the operational constraints of the deployment environment, balancing raw accuracy against throughput and resource availability.

10.3 Industrial Implications and Deployment Challenges

The empirical success of the machine learning models demonstrated in this thesis, particularly the 2D xresnet34 and the efficient TCN, provides the technical foundation for a paradigm shift in on-line steel surface metrology. However, transitioning these models from an experimental setting to a live production environment involves specific industrial opportunities and deployment challenges.

10.3.1 Potential Industrial Applications

The primary and immediate application of this research is the enablement of continuous process monitoring for the prevention of coil value loss or product downgrading. Currently, operators must wait until a coil is finished to take the head and tail stylus measurements. By deploying the R_a prediction models developed in this thesis, line operators receive a continuous, real-time roughness estimate across the entire strip length. This provides an early warning system for incorrectly specified steel, allowing operators to intervene mid-coil rather than discovering the error only after the coil is complete. Crucially, the physical stylus would remain the requisite standard for customer release, serving as a ground-truth anchor to validate the model's predictions and detect potential concept drift. Simultaneously, the generated full-coil roughness profile could be provided to customers as a significant value-added service, offering them a level of quality assurance previously unavailable in the market.

As trust in the system matures, these models serve as the enabling sensor for Closed-Loop Roughness Control. The low inference latency of the TCN architecture allows for feedback within the mill's control cycle. This enables a fully automated loop where parameters such as roll force, tension, and texture transfer efficiency are dynamically adjusted to maintain the target roughness specification, systematically reducing the production of off-specification material.

10.3.2 Challenges to Deployment

Despite the predictive accuracy achieved, several barriers remain to full industrial integration:

- **Data Drift, Domain Shift, and Sensor Degradation:** The models in this thesis rely on labelled data captured in a controlled laboratory environment and unlabelled pretraining data from a specific production window. In a continuous production environment, the physical degradation of optical sensors (e.g., lens fouling, laser intensity fade) or shifts in the steel grade mix over time may cause the live input data distribution to drift from these training distributions. To maintain accuracy, a robust MLOps pipeline is required to monitor prediction confidence and trigger periodic retraining using the routine head and tail stylus measurements as a validation anchor.

- **Software Interoperability and Automation:** A significant practical hurdle is the integration of the model with the existing commercial software. Currently, the proprietary on-line laser measurement system requires manual intervention to export raw intensity readings, which prevents real-time automated analysis. Successful deployment requires the development of a seamless data pipeline, either through vendor-supported APIs or automated bot scripts to simulate manual extraction, to feed raw sensor data to the model and present the predicted R_a to operators via an accessible dashboard.

- **Trust and Interpretability:** Unlike physical stylus measurements, deep learning models operate as black boxes. For operators to trust the system enough to automate control decisions, further work on model interpretability, such as visualising which specific surface texture features trigger a high R_a prediction, will be necessary to distinguish between true roughness and transient surface artefacts (e.g., oil or dust).

- **Latency vs. Line Speed:** While Chapter 8 confirmed that 1D models like TCNs are highly efficient, the trade-off between model capacity and inference speed remains critical. The inference time must remain strictly lower than the data acquisition rate to prevent data backlogs, potentially favouring lighter models (like the TCN ks=7, nl=8) over slightly more accurate but heavier 2D architectures (like xresnet34).

- **Model Versioning and Calibration Consistency:** A data-driven model represents a learned mapping from sensor input to predicted R_a. Modifying the architecture, retraining on new data, or deploying a different model version is conceptually analogous to

updating an analytical formula; in either case, quantitative outputs for identical raw inputs may change. In the present industrial context, the objective of the model is fast feedback control rather than product certification, which remains governed by stylus measurements. Consequently, absolute numerical invariance is less critical than maintaining stable predictions within acceptable process tolerance bands. Nevertheless, deployment requires formal model governance procedures. Any updated model must be validated against routine stylus measurements to ensure continued calibration to the industry-standard R_a. Version control, validation protocols, and controlled update schedules are therefore essential components of a production-ready system. This requirement reflects an operational consideration common to both analytical and learned approaches and is necessary for long-term industrial integration.

10.4 Future Directions

The work of Chapter 6 introduced several areas which can be explored further in the future. Although we developed a robust multi-metric approach for identifying poor-quality data, which was validated through visual analysis of several data samples, it remains an open question whether this is a process which can also be improved through the use of machine learning. Several of the models we applied in subsequent chapters, such as a TCN or autoencoder, could be adapted for this task, and we have already demonstrated that the TCN is a promising approach when working with our steel surface data. Automating the identification of the data would allow us to construct an end-to-end pipeline, where trained models are able to identify gaps and fill them in a single process. Furthermore, the imputation models developed in this chapter offer another promising avenue for future investigation. These models were trained specifically to reconstruct missing segments in high-fidelity stylus data by learning to preserve critical statistical properties. A valuable next step would be to apply these trained models in a zero-shot transfer setting to the profiles generated by the closed-form solution from the on-line laser sensor. This would allow us to directly assess whether using our models to 'clean' the laser-derived profiles improves their correlation with the gold-standard stylus measurements, potentially providing a practical method for enhancing the accuracy of the on-line system. One final addition to our existing work could be to extend the statistics-aware loss function to include other relevant surface roughness parameters such as R_{sk} (Skewness) or R_q (Root Mean Square Roughness). This may improve the robustness of the reconstruction model, and possibly prevent overfitting by incorporating a more global view of the surface roughness.

Additionally, the work in Chapter 6 could be progressed in a different direction by replacing the handcrafted heuristic functions used for poor-segment identification with a learned missingness detection model. Rather than manually defining thresholds based on intensity statistics, a neural network could be trained to predict segment reliability directly from raw laser data. This could be achieved by simulating ground-truth surface profiles, emulating stylus–laser discrepancies, or formulating a domain adaptation objective that aligns reconstructed profiles with trusted stylus-derived statistics. Such an approach would allow the missingness function itself to be optimised via backpropagation, potentially yielding a more principled and generalisable solution than manually engineered metrics, while also improving interpretability of the conditions under which the laser-based measurement system fails.

While Chapter 7 demonstrated the computational efficiency of the ROCKET framework, our attempts to fine-tune the model highlighted fundamental architectural constraints that warrant further theoretical investigation. Unlike deep neural networks that benefit from hierarchical feature extraction, ROCKET relies on a single, flat projection layer using fixed random kernels. This shallow architecture may inherently limit the model's capacity to learn the complex, composite features necessary for further accuracy gains via backpropagation. Furthermore, for our training from pruned kernel experiments, there is a reliance on fixed initialisation which might be creating a static loss landscape; initialising training from this state may situate the model in a sub-optimal local minimum from which gradient descent cannot easily escape. Future work could investigate alternative initialisation strategies or hybrid architectures that introduce learnable non-linearities earlier in the pipeline. Additionally, the hard non-linearity of the PPV pooling mechanism and the saturation of sigmoid activations likely contribute to vanishing gradients, impeding effective fine-tuning. Investigating differentiable approximations of these pooling functions or alternative activation constraints could unlock the potential for end-to-end optimisation of ROCKET-based models.

All of the models and the tasks we explored in Chapters 6, 7 and 8 can be expanded to wider domains. For example, our optimised, fine-tuned version of the ROCKET model could now be tested on other high-speed industrial monitoring tasks, while the most effective models in our R_a prediction experiments could be employed to predict a wider spectrum of steel surface parameters beyond just R_a.

Furthermore, the distinct methodologies developed for surface profile reconstruction (Chapter 6) and the efficient ROCKET framework (Chapter 7) represent contributions cur-

rently being prepared for publication, highlighting their potential broader impact.

Additionally, the work of Chapter 9 offers some of the most promising directions for future work. Although the pretraining approaches we investigated showed promise, there is room for improvement, specifically by investigating methods for reducing variability in consecutive runs of the approach. We have also demonstrated the importance of honing the pretraining approach for the end-task through our coil classification approach, suggesting that hybrid methods which combine pretraining objectives more directly aligned with the regression task is a promising avenue for other domains as well. In particular, for our setting, it could be beneficial to investigate a dual-task learning approach, where the model aligns coils in the representation space in line with their R_a value by incorporating R_a with the labelled data during pretraining. This approach offers the opportunity to align coils with respect to their R_a value, while increasing generalisation to new coils, and represents a promising direction for future research.

An important limitation of the representation learning experiments in this thesis is that pretraining was applied only to the computationally efficient TCN ks=7, nl=8 backbone. The strongest raw predictive model identified in Chapter 8, namely the xresnet34 architecture, was trained only from scratch due to computational constraints. A natural extension of this work would therefore be to apply the same pretraining strategies (e.g., coil classification or contrastive learning) to the higher-capacity 2D convolutional architectures. It is plausible that combining large-scale pretraining with these more expressive models would yield the strongest overall predictive performance. Furthermore, evaluating such pretrained architectures under the leave-one-steel-sample-out cross-validation framework would allow direct assessment of whether representation learning improves generalisation to previously unseen steel coils. Such experiments require substantially greater computational resources and training time, but represent a clear opportunity for further improvement.

10.4.0.1 Funding and License

This work was funded by EPSRC Industrial Case award (EP/V519601/1).

Bibliography

[1] Geometrical product specifications (gps) – surface texture: Profile method – rules and procedures for the assessment of surface texture, 1996. https://www.iso.org/standard/13693.html, accessed 2025-09-22.

[2] Geometrical product specifications (gps) – filtration – part 21: Gaussian filter – linear profile filters, 2011. https://www.iso.org/standard/51218.html, accessed 2025-09-22.

[3] M. M. Abdulrazzaq, N. T. Ramaha, A. A. Hameed, M. Salman, D. K. Yon, N. L. Fitriyani, M. Syafrudin, and S. W. Lee. Consequential advancements of self-supervised learning (ssl) in deep learning contexts. *Mathematics*, 12(5):758, 2024.

[4] A. Ahmad, E. J. Rose, M. Roy, and E. Valachovic. Enhancing data completeness in time series: Imputation strategies for missing data using significant periodically correlated components. *arXiv preprint arXiv:2505.02008*, 2025.

[5] Alinco Belgium. Surface roughness values – stainless steel finishing and post-treatment. `https://alinco.be/en/materials/stainless-steel/stainless-steel-finishing-and-post-treatment/surface-roughness-values`, 2025. Accessed: 2025-09-22.

[6] ArcelorMittal. Magnelis®: The steel coating that offers the best corrosion resistance. Technical report, ArcelorMittal Commercial Sections S.A., 2014. Accessed on September 10, 2025. Available at: https://industry.arcelormittal.com/repository2/About/Magnelis.pdf.

[7] A. Bagnall, M. Flynn, J. Large, J. Lines, and M. Middlehurst. On the usage and performance of the hierarchical vote collective of transformation-based ensembles version 1.0. In *2nd Workshop on Advanced Analytics and Learning on Temporal Data*. INRIA, 2020.

[8] S. Bai, J. Z. Kolter, and V. Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[9] P. Baldi. Autoencoders, unsupervised learning, and deep architectures. *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, 2012. Originally introduced by Rumelhart et al., 1986.

[10] H. Bao, L. Dong, S. Piao, and F. Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[11] A. F. Bastawros, J. G. Speer, G. Zerafa, and R. P. Krupitzer. Effects of steel surface texture on appearance after painting. Technical report, SAE Technical Paper, 1993.

[12] W. Bilstein, W. Enderle, G. Moreas, D. Oppermann, T. Routschek, and F. Van De Velde. Two systems for on-line oil film and surface roughness measurement for strip steel production. *Metallurgical Research & Technology*, 104(7-8):348–353, 2007.

[13] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, 2006.

[14] L. Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[15] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[16] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li. Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31, 2018.

[17] S. Chen, W. Sun, L. Huang, X. P. Li, Q. Wang, and D. John. Pocket: Pruning random convolution kernels for time series classification from a feature selection perspective. *Knowledge-Based Systems*, 300:112253, 2024.

[18] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[19] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[20] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 1597–1607, 2020.

[21] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[22] X. Chen, M. Ding, X. Wang, Y. Xin, S. Mo, Y. Wang, S. Han, P. Luo, G. Zeng, and J. Wang. Context autoencoder for self-supervised representation learning. *International Journal of Computer Vision*, 132(1):208–223, 2024.

[23] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[24] X. Chen, C. Liang, D. Huang, E. Real, K. Wang, Y. Liu, H. Pham, X. Dong, T. Luong, C.-J. Hsieh, et al. Symbolic discovery of optimization algorithms. *arXiv preprint arXiv:2302.06675*, 2023.

[25] S.-H. Cheri, J.-G. Wang, and T.-Q. Gu. An online intelligent control method for surface roughness of cold-rolled strip steel. In *2018 37th Chinese Control Conference (CCC)*, pages 8330–8335. IEEE, 2018.

[26] S.-H. Cheri, J.-G. Wang, and T.-Q. Gu. An online intelligent control method for surface roughness of cold-rolled strip steel. In *2018 37th Chinese Control Conference (CCC)*, pages 8330–8335, 2018.

[27] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

[28] A. Dempster, F. Petitjean, and G. I. Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

[29] A. Dempster, D. F. Schmidt, and G. I. Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 248–257, 2021.

[30] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[32] R. P. Dos Santos, J. P. Matos-Carvalho, and V. R. Leithardt. Deep learning in time series forecasting with transformer models and rnns. *PeerJ Computer Science*, 11:e3001, 2025.

[33] L. dos Santos Coelho, H. V. Hultmann Ayala, and V. Cocco Mariani. Co and nox emissions prediction in gas turbine using a novel modeling pipeline based on the combination of deep forest regressor and feature engineering. *Fuel*, 355:129366, 2024.

[34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[35] W. Du, D. Côté, and Y. Liu. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023.

[36] J. Duan, W. Zheng, Y. Du, W. Wu, H. Jiang, and H. Qi. Mf-clr: multi-frequency contrastive learning representation for time series. In *Forty-first International Conference on Machine Learning*, 2024.

[37] M. Elangovan, N. Sakthivel, S. Saravanamurugan, B. Nair, and V. Sugumaran. Machine learning approach to the prediction of surface roughness using statistical features of vibration signal acquired in turning. *Procedia Computer Science*, 50:282–288, 2015. Big Data, Cloud and Computing Challenges.

[38] Elexis Group. Emg sorm® roughness and waviness measurement according to industry standards. `https://www.elexis.group/en/news/news-detail/emg-sormr-roughness-and-waviness-measurement-according-to-industry-standard` 2025. Accessed: 2025-09-22.

[39] W. Emmens. The influence of surface roughness on friction. *IDDRG Congress, Dearborn, USA, May 16-18*, Proceedings 15 th.:63–70, 1988.

[40] X. Feng, X. Gao, and L. Luo. A resnet50-based method for classifying surface defects in hot-rolled strip steel. *Mathematics*, 9(19):2359, 2021.

[41] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The annals of mathematical statistics*, 11(1):86–92, 1940.

[42] S. Gao, T. Koker, O. Queen, T. Hartvigsen, T. Tsiligkaridis, and M. Zitnik. Units: A unified multi-task time series model. *Advances in Neural Information Processing Systems*, 37:140589–140631, 2024.

[43] S. Gao, K. Yang, H. Shi, K. Wang, and J. Bai. Review on panoramic imaging and its applications in scene understanding. *IEEE Transactions on Instrumentation and Measurement*, 71:1–34, 2022.

[44] Global Energy Monitor. Tata steel port talbot steel plant. `https://www.gem.wiki/Tata_Steel_Port_Talbot_steel_plant`, June 2025. Accessed: 2025-09-22.

[45] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, Cambridge, MA, 2016.

[46] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[47] J. Gui, T. Chen, J. Zhang, Q. Cao, Z. Sun, H. Luo, and D. Tao. A survey on self-supervised learning: Algorithms, applications, and future trends. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(12):9052–9071, Dec. 2024.

[48] A. K. Gupta, S. Chakroborty, S. K. Ghosh, and S. Ganguly. A machine learning model for multi-class classification of quenched and partitioned steel microstructure type by the k-nearest neighbor algorithm. *Computational Materials Science*, 228:112321, 2023.

[49] H. Han, M. Sun, H. Han, X. Wu, and J. Qiao. Univariate imputation method for recovering missing data in wastewater treatment process. *Chinese Journal of Chemical Engineering*, 53:201–210, 2023.

[50] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, 2020.

[51] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[52] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[53] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 558–567, 2019.

[54] G. Henderson. Steel company of wales / british steel corporation production system computer projects at abbey works, port talbot 1965-1973: A brief history. Technical report, Archives IT, 2019.

[55] J. Howard et al. fastai. `https://github.com/fastai/fastai`, 2018.

[56] C.-L. Hwang and K. Yoon. *Multiple Attribute Decision Making: Methods and Applications A State-of-the-Art Survey*, volume 186 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, Berlin, Heidelberg, 1981.

[57] International Organization for Standardization. Geometrical product specifications (gps) – surface texture: Profile method – nominal characteristics of contact (stylus) instruments, 1996.

[58] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[59] H. Ismail Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.

[60] A. Jawaid, S. Schmidt, M. Lotz, and J. Seewig. Surface data imputation with stochastic processes. *Measurement Science and Technology*, 36(4):046131, 2025.

[61] D. Jiang, H. Yang, H. Cao, and D. Xu. A missing data imputation method for industrial soft sensor modeling. *Journal of Process Control*, 152:103485, 2025.

[62] H. Kang and S. Choi. Bayesian common spatial patterns for multi-subject eeg classification. *Neural Networks*, 57:39–50, 2014.

[63] M. Kazijevs and M. D. Samad. Deep imputation of missing values in time series health data: A review with benchmarking. *Journal of biomedical informatics*, 144:104440, 2023.

[64] M. Khan, H. Wang, A. Ngueilbaye, and A. Elfatyany. End-to-end multivariate time series classification via hybrid deep learning architectures. *Personal and Ubiquitous Computing*, pages 1–15, 2020.

[65] E. A. Kholief, S. H. Darwish, and M. N. Fors. Detection of steel surface defect based on machine learning using deep auto-encoder network. *Industrial engineering and operations management*, pages 218–229, 2017.

[66] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[67] C. M. Köllőd, A. Adolf, K. Iván, G. Márton, and I. Ulbert. Deep comparisons of neural networks from the eegnet family. *Electronics*, 12(12):2743, 2023.

[68] J. Koo, S. Lee, A. M. C. Baek, E. Park, and N. Kim. Downskin surface roughness prediction with machine learning for as-built cm247lc fabricated via powder bed fusion with a laser beam. *3D Printing and Additive Manufacturing*, 11(4):1510–1522, 2024.

[69] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105. Curran Associates Inc, 2012.

[70] I. La Fé-Perdomo, J. Ramos-Grez, R. Mujica, and M. Rivas. Surface roughness ra prediction in selective laser melting of 316l stainless steel by means of artificial intelligence inference. *Journal of King Saud University-Engineering Sciences*, 35(2):148–156, 2023.

[71] I. La Fé-Perdomo, J. Ramos-Grez, R. Mujica, and M. Rivas. Surface roughness ra prediction in selective laser melting of 316l stainless steel by means of artificial intelligence inference. *Journal of King Saud University - Engineering Sciences*, 35(2):148–156, 2023.

[72] N. P. Lawrence, S. K. Damarla, J. W. Kim, A. Tulsyan, F. Amjad, K. Wang, B. Chachuat, J. M. Lee, B. Huang, and R. Bhushan Gopaluni. Machine learning for industrial sensing and control: A survey and practical perspective. *Control Engineering Practice*, 145:105841, 2024.

[73] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.

[74] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[75] Z. Li, S. Song, Z. Wu, X. Hu, B. Wang, and S. Zhang. Basic oxygen steelmaking slag: Formation, reaction, and energy and material recovery. *Steel Research International*, 92(7):2100167, 2021.

[76] W. Liu, C. Li, M. M. Rahamana, T. Jiang, H. Sun, X. Wu, W. Hu, H. Chen, C. Sun, Y. Yao, and M. Grzegorzek. Is the aspect ratio of cells important in deep learning? a robust comparison of deep learning methods for multi-scale cytopathology cell image classification: from convolutional neural networks to visual transformers, 2021.

[77] Z. Liu, H. Hu, Y. Lin, Z. Yao, Z. Xie, Y. Wei, J. Ning, Y. Cao, Z. Zhang, L. Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12009–12019, 2022.

[78] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022.

[79] F. Luk, V. Huynh, and W. North. Measurement of surface roughness by a machine vision system. *Journal of physics E: Scientific instruments*, 22(12):977, 1989.

[80] T. maintainers and contributors. Torchvision: Pytorch's computer vision library. `https://github.com/pytorch/vision`, 2016.

[81] P. Malhotra, V. TV, L. Vig, P. Agarwal, and G. Shroff. Timenet: Pre-trained deep recurrent neural network for time series classification. *arXiv preprint arXiv:1706.08838*, 2017.

[82] D. Mazumdar and J. W. Evans. *Modeling of Steelmaking Processes*. CRC Press, 2009.

[83] A. Meena and P. Ramana. Highly nonlinear mathematical regression for polypropylene material strength prognostication. *Materials Today: Proceedings*, 62:696–702, 2022. Second International Conference on Engineering Materials, Metallurgy and Manufacturing.

[84] M. Middlehurst, J. Large, A. Bagnall, M. Flynn, and J. Lines. Hive-cote 2.0: a new meta ensemble for time series classification. In *Proceedings of the KDD Workshop on Mining and Learning from Time Series (MiLeTS)*. KDD, 2021.

[85] A. Milne and X. Xie. Steel surface roughness parameter calculations using lasers and machine learning models. *The International Journal of Advanced Manufacturing Technology*, 2024.

[86] A. Milne, X. Xie, and G. K. L. Tam. Pretraining techniques for r_a prediction with long thin spatial industrial data. In *Proceedings of Advanced Concepts for Intelligent Vision Systems*, 2025. Accepted for presentation, to appear.

[87] A. J. M. Milne and X. Xie. Predicting surface texture in steel manufacturing at speed, 2023.

[88] D. D. Mohite, V. S. Jadhav, A. N. Nayak, and S. S. Chavan. An influence of cnc grinding wheel dressing parameters on ra value of en19 steel. *Materials Today: Proceedings*, 2023.

[89] R. T. Mushtaq, A. Iqbal, Y. Wang, A. M. Khan, and M. S. Abu Bakar. Parametric optimization of 3d printing process hybridized with laser-polished petg polymer. *Polymer Testing*, 125:108129, 2023.

[90] R. T. Mushtaq, Y. Wang, A. M. Khan, M. Rehman, X. Li, and S. Sharma. A post-processing laser polishing method to improve process performance of 3d printed new industrial nylon-6 polymer. *Journal of Manufacturing Processes*, 101:546–560, 2023.

[91] T. S. Nederland. Process of steelmaking. `https://www.tatasteelnederland.com/en/How-we-make-steel/process-of-steelmaking`, 2025. Accessed September 11, 2025.

[92] S. D. L. Nogueira et al. Prediction of the nox and co2 emissions from an experimental dual fuel engine using optimized random forest combined with feature engineering. *Energy*, 280:128066, 2023.

[93] I. Oguiza. tsai - a state-of-the-art deep learning library for time series and sequential data. Github, 2022.

[94] Olympus Corporation. *Introduction to Surface Roughness Measurement: Roughness Measurement Guidebook*, 2018. Measuring Laser Microscope OLS5000, LEXT Series.

[95] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[96] Y. S. Perera, D. Ratnaweera, C. H. Dasanayaka, and C. Abeykoon. The role of artificial intelligence-driven soft sensors in advanced sustainable process industries: A critical review. *Engineering Applications of Artificial Intelligence*, 121:105988, 2023.

[97] V. C. Pezoulas, K. D. Kourou, E. Mylona, C. Papaloukas, A. Liontos, D. Biros, O. I. Milionis, C. Kyriakopoulos, K. Kostikas, H. Milionis, and D. I. Fotiadis. Icu admission and mortality classifiers for covid-19 patients based on subgroups of dynamically associated profiles across multiple timepoints. *Computers in Biology and Medicine*, 141:105176, 2022.

[98] Y. Qiao, J. Lü, T. Wang, K. Liu, B. Zhang, and H. Snoussi. A multihead attention self-supervised representation model for industrial sensors anomaly detection. *IEEE Transactions on Industrial Informatics*, 20(2):2190–2199, 2023.

[99] E. Rahimian, S. Zabihi, S. F. Atashzar, A. Asif, and A. Mohammadi. Xceptiontime: A novel deep architecture based on depthwise separable convolutions for hand gesture classification. *arXiv preprint arXiv:1911.03803*, 2019.

[100] L. Ren, T. Wang, Y. Laili, and L. Zhang. A data-driven self-supervised lstm-deepfm model for industrial soft sensor. *IEEE Transactions on Industrial Informatics*, 18(9):5859–5869, 2021.

[101] A. P. Ruiz, M. Flynn, J. Large, M. Middlehurst, and A. Bagnall. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 35(2):401–449, 2021.

[102] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[103] H. Salehinejad, Y. Wang, Y. Yu, T. Jin, and S. Valaee. S-rocket: Selective random convolution kernels for time series classification. *arXiv preprint arXiv:2203.03445*, 2022.

[104] L. Sanchez, N. Costa, and I. Couso. Data imputation in the frequency domain using echo state networks. *Engineering Applications of Artificial Intelligence*, 144:110129, 2025.

[105] T. Saric, G. Simunovic, and K. Simunovic. Use of neural networks in prediction and simulation of steel surface roughness. *International journal of simulation modelling*, 12(4):225–236, 2013.

[106] K. Sawai, T.-T. Chen, F. Sun, T. Ogawa, and Y. Adachi. Image regression analysis for linking the microstructure and property of steel. *Results in Materials*, 21:100526, 2024.

[107] A. Sayer. New developments in manufacturing: the just-in-time system. *Capital & Class*, 10(3):43–72, 1986.

[108] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball. Deep learning with convolutional neural networks for eeg decoding and visualization. *Human brain mapping*, 38(11):5391–5420, 2017.

[109] S. Seetharaman. *Treatise on Process Metallurgy: Industrial Processes*. Elsevier, 2014.

[110] M. Sharp, R. Ak, and T. Hedberg Jr. A survey of the advancing use and development of machine learning in smart manufacturing. *Journal of manufacturing systems*, 48:170–179, 2018.

[111] M. Soori, B. Arezoo, and R. Dastres. Internet of things for smart factories in industry 4.0, a review. *Internet of Things and Cyber-Physical Systems*, 3:192–204, 2023.

[112] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[113] C. W. Tan, C. Bergmeir, F. Petitjean, and G. I. Webb. Time series extrinsic regression: Predicting numeric values from time series data. *Data Mining and Knowledge Discovery*, 35:1032–1060, 2021.

[114] C. W. Tan, A. Dempster, C. Bergmeir, and G. I. Webb. Multirocket: multiple pooling operators and transformations for fast and effective time series classification. *Data Mining and Knowledge Discovery*, 36(5):1623–1646, 2022.

[115] Y. Tashiro, J. Song, Y. Song, and S. Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in neural information processing systems*, 34:24804–24816, 2021.

[116] Tata Steel UK. Plans approved for electric arc furnace at port talbot. `https://www.tatasteeluk.com/corporate/news/plans-approved-for-electric-arc-furnace`, February 2025. Accessed: 2025-09-22.

[117] TJ Sky Steel. What are the surface roughness requirements for high carbon steel plate? `https://www.tjskysteel.com/blog/what-are-the-surface-roughness-requirements-for-high-carbon-steel-plate-100.html`, 2025. Accessed: 2025-09-22.

[118] D. H. Tran, V. L. Nguyen, H. Nguyen, and Y. M. Jang. Self-supervised learning for time-series anomaly detection in industrial internet of things. *Electronics*, 11(14):2146, 2022.

[119] G. Uribarri, F. Barone, A. Ansuini, and E. Fransén. Detach-rocket: sequential feature selection for time series classification with random convolutional kernels. *Data Mining and Knowledge Discovery*, 38(6):3922–3947, 2024.

[120] M. Vishnu Vardhan, G. Sankaraiah, and M. Yohan. Optimization of cutting parameters and prediction of ra & mrr for machining of p20 steel on cnc milling using artificial neural networks. *Materials Today: Proceedings*, 5(13, Part 3):27058–27064, 2018. International Conference on Advances in Materials and Manufacturing, December 8-10, 2016.

[121] Z. Wang, W. Yan, and T. Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1585, 2017.

[122] S. C. Wetstein, V. M. de Jong, N. Stathonikos, M. Opdam, G. M. Dackus, J. P. Pluim, P. J. van Diest, and M. Veta. Deep learning-based breast cancer grading and survival analysis on whole-slide histopathology images. *Scientific reports*, 12(1):15102, 2022.

[123] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.

[124] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9653–9663, 2022.

[125] H. Yang et al. Optimization of tight gas reservoir fracturing parameters via gradient boosting regression modeling. *Heliyon*, 2024.

[126] L. Yang and S. Hong. Unsupervised time-series representation learning with iterative bilinear temporal-spectral fusion. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 25038–25054. PMLR, 17–23 Jul 2022.

[127] X. Yang, Z. Zhang, and R. Cui. Timeclr: A self-supervised contrastive learning framework for univariate time series representation. *Knowledge-Based Systems*, 245:108606, 2022.

[128] J. Yoon, D. Jarrett, and M. Van der Schaar. Time-series generative adversarial networks. *Advances in neural information processing systems*, 32, 2019.

[129] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 8980–8987, 2022.

[130] S. Zhang, T. Yu, B. Choi, F. Ouyang, and Z. Ding. Radiomap inpainting for restricted areas based on propagation priority and depth map. *IEEE Transactions on Wireless Communications*, 23(8):9330–9344, 2024.

[131] X. Zhang, Z. Zhao, T. Tsiligkaridis, and M. Zitnik. Self-supervised contrastive pre-training for time series via time-frequency consistency. *Advances in neural information processing systems*, 35:3988–4003, 2022.

[132] N. Zhao, Z. Wu, R. W. H. Lau, and S. Lin. What makes instance discrimination good for transfer learning?, 2021.

[133] X. Zheng, T. Wang, W. Cheng, A. Ma, H. Chen, M. Sha, and D. Luo. Parametric augmentation for time series contrastive learning. *arXiv preprint arXiv:2402.10434*, 2024.

[134] X. Zheng, S. Zheng, Y. Kong, and J. Chen. Recent advances in surface defect inspection of industrial products using deep learning techniques. *The International Journal of Advanced Manufacturing Technology*, 113(1):35–58, 2021.

[135] X. Zou, Z. Wang, Q. Li, and W. Sheng. Integration of residual network and convolutional neural network along with various activation functions and global pooling for time series classification. *Neurocomputing*, 367:39–45, 2019.