FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

A Machine Learning Approach to Tool Condition Monitoring in Drilling

André Ribeiro Oliveira



Master's Degree in Mechanical Engineering

Supervisor: Prof. Armando Jorge Miranda de Sousa

Co-supervisor: Prof. Ana Rosanete Lourenço Reis

September 22, 2026

A Machine Learning Approach to Tool Condition Monitoring in Drilling

André Ribeiro Oliveira

Master's Degree in Mechanical Engineering

Approved in oral examination by the committee:

President: Prof. António Manuel Ferreira Mendes Lopes

Examiner: Prof. Mafalda Pereira Gonçalves

Member: Prof. Armando Jorge Miranda de Sousa

September 22, 2026

© André Ribeiro Oliveira, 2026

Resumo

O desgaste de ferramentas é um dos principais fatores que limitam a maquinagem moderna. Afeta diretamente a qualidade das peças, os custos operacionais e a segurança do processo. Na furação, este problema é ainda mais crítico, pois a operação costuma ocorrer nas etapas finais do fabrico, em que uma falha da ferramenta pode causar a rejeição de componentes já maquinados de alto valor.

Este trabalho tem como objectivo o desenvolvimento e a avaliação de um sistema de monitorização do estado da ferramenta (TCM, do inglês *Tool Condition Monitoring*) para operações de furação em aço inoxidável AISI 316L baseado em métodos de medição indireta e em técnicas baseadas em dados. É usada uma configuração com vários sensores, formada por um transdutor de força de mesa, um acelerómetro triaxial, um microfone e dois sensores de corrente que coletam simultaneamente forças, vibração, som e corrente do motor durante a maquinação.

Foi realizada uma campanha experimental fatorial completa com nove condições de corte, variando o avanço e a velocidade de rotação da ferramenta em torno dos valores nominais recomendados pelo fabricante. O desgaste de flanco (V_B) foi medido periodicamente com um microscópio digital e utilizado como referência (*ground truth*) para dividir os dados em três classes.

A cadeia de processamento inclui a segmentação por ciclo de furação (*peck*), a redução de ruído por subtração espectral, a extração de características nos domínios do tempo e da frequência e estratégias de tratamento do desequilíbrio de classes. Foram comparados três modelos de classificação: Random Forest e XGBoost, como referência, e uma rede neuronal convolucional (CNN) unidimensional multiramo, que opera diretamente sobre os sinais brutos dos sensores, com ramos independentes que processam cada grupo de sensores em paralelo, de modo a lidar com as diferentes frequências de amostragem do sistema de aquisição.

Os modelos foram avaliados de duas formas: uma divisão *leave-one-out* ao nível da experiência, que testa a generalização para condições de corte nunca vistas durante o treino, e uma divisão ao nível do furo, que avalia o desempenho em condições já conhecidas. Em condições conhecidas, a CNN obteve o melhor desempenho, com um macro-F1 de 90,19%, face a 88,13% do XGBoost e 81,40% do Random Forest. Em condições nunca vistas, a ordem inverte-se: o XGBoost é o que melhor generaliza (63,26%), seguido de perto pelo Random Forest (61,69%), enquanto a CNN desce para 59,48%, mais de 30 pontos percentuais abaixo do seu desempenho em condições conhecidas.

Foi ainda realizado um estudo de ablação segundo cada um destes eixos: seleção de *pecks*, conjunto de sensores/canais, estratégia de tratamento do desequilíbrio de classes e dimensão da janela de entrada, de modo a isolar o contributo individual de cada opção de projeto.

Keywords: Desgaste da Ferramenta • Maquinagem • Fusão de sensores • Inteligência Artificial • Aço Inoxidável

Abstract

Tool wear is one of the primary limiting factors in modern machining, directly affecting part quality, operational costs, and process safety. In drilling operations, this problem is particularly critical since drilling is typically performed at the final stages of manufacturing, where tool failure can result in rejecting fully machined, high-value components.

This work presents the development and evaluation of a Tool Condition Monitoring (TCM) system for drilling operations in AISI 316L stainless steel, based on indirect sensing methods and data driven techniques. The system employs a multi-sensor setup comprising a table dynamometer, a three-axis accelerometer, a microphone, and two current sensors, simultaneously acquiring force, vibration, acoustic, and motor current signals during machining.

A full factorial experimental campaign was conducted with nine cutting conditions, varying feed rate and spindle speed around the nominal values recommended by the tool manufacturer. Flank wear (V_B) was measured periodically using a digital microscope and used as ground truth for dividing the data into three classes.

The processing pipeline includes peck segmentation, spectral subtraction denoising, time and frequency-domain feature extraction, and class imbalance handling strategies. Three classification models were compared: a Random Forest and an XGBoost classifier as baselines, and a multi-branch one-dimensional Convolutional Neural Network (CNN) operating directly on raw sensor signals, with independent branches processing each sensor group in parallel to handle the heterogeneous sampling rates of the acquisition system.

The models were evaluated in two ways: a file-level leave-one-out split, which tests generalisation to cutting conditions never seen during training, and a hole-level split, which evaluates performance within already-known conditions. Within known conditions, the CNN came out on top, with a macro-F1 of 90.19%, against 88.13% for XGBoost and 81.40% for Random Forest. Move to conditions the models have never seen, and the order flips: XGBoost generalises best (63.26%), Random Forest sits close behind (61.69%), and the CNN drops to 59.48% over 30 points off its in-distribution score.

An ablation study was also carried out along each of these axes: peck selection, sensor/channel set, class-imbalance strategy, and input window size to isolate the individual contribution of each design choice.

Keywords: Tool Wear • Machining • Sensor Fusion • Artificial Intelligence • Stainless steel

The United Nations Sustainable Development Goals

The United Nations Sustainable Development Goals (SDGs) provide a global framework for promoting sustainable development. This dissertation is primarily related to SDG 9 (Industry, Innovation and Infrastructure) and SDG 12 (Responsible Consumption and Production), through the development of a data-driven Tool Condition Monitoring framework for drilling operations.

The work aligns with SDG 9, particularly Target 9.5, by developing and evaluating a TCM system based on indirect sensing, multi-sensor data, signal processing, and machine learning. The combination of force, vibration, acoustic, and spindle-current signals contributes to the development of data-driven technologies for industrial applications.

The work also relates to Target 9.4, which promotes resource-use efficiency and more sustainable industrial processes. By monitoring tool degradation, the proposed system can support more informed decisions regarding tool utilisation and replacement. However, the dissertation does not directly quantify improvements in resource or environmental efficiency.

Regarding SDG 12, Target 12.2 is relevant because more informed tool replacement decisions may contribute to a more efficient use of cutting tools and manufacturing resources. Similarly, monitoring tool degradation may potentially help prevent tool-related defects and unnecessary replacement, with possible implications for reducing production waste. These potential contributions were not directly measured in the present study.

Overall, the contribution of the dissertation to the SDGs lies in the development of an innovative, data-driven approach to tool condition monitoring, with potential to support more efficient use of manufacturing resources and improved decision-making in machining processes.

SDG	Target	Contribution of the dissertation
9	9.4 - Increased resource use efficiency	Tool condition monitoring can potentially support more efficient tool utilisation and more informed replacement decisions.
	9.5 - Enhance scientific research and technological capabilities	Development of a data-driven TCM framework combining multi-sensor data, signal processing, and machine learning for industrial machining.
12	12.2 - Sustainable management and efficient use of natural resources	The proposed monitoring approach can potentially support more efficient use of cutting tools and manufacturing resources.

Acknowledgements

I want to express my sincere gratitude to the supervisory team for their guidance, which allowed this work to progress smoothly and considerably lightened the demands of the process. I would especially like to thank my supervisor, Professor Armando Sousa, whose guidance and support throughout this dissertation were truly essential.

I am equally grateful to José Ferreira for his genuine interest in helping me produce more rigorous work. Our discussions and his constant availability made this journey far more rewarding than it would otherwise have been.

My sincere thanks also go to Professor Tiago Silva, whose valuable insight and constant willingness to help greatly contributed to the smooth running of the experimental work.

I would also like to acknowledge the technical staff at Oficinas de Mecânica for their assistance in setting up the experimental apparatus, and to thank Palbit for kindly supplying the solid carbide drills and workpieces used throughout this dissertation.

To my friends, thank you for your support, particularly those I had the privilege of meeting during my master's degree, whose companionship made this journey considerably more enjoyable.

To my girlfriend, my heartfelt thanks for standing by my side throughout this journey and for the strength and encouragement she so steadfastly provided.

Finally, and by no means least, I wish to thank my family, to whom I owe the very possibility of this achievement, for their unconditional and unwavering support.

André Ribeiro Oliveira

Institutional acknowledgements

This research was funded by COMPETE 2030, project “SNext (Nova geração de ferramentas híbridas)”, grant number COMPETE2030-FEDER-00582100.

The author also gratefully acknowledges funding from Project Hi-rEV (“Recuperação do Setor de Componentes Automóveis”, C644864375-00000002), cofinanced by Plano de Recuperação e Resiliência (PRR), República Portuguesa, through NextGeneration EU.



Declaration of honour

This dissertation is the result of my own work and has not been submitted for assessment at this or any other institution. Any statement, idea or result drawn from other authors is properly attributed, both within the text and in the bibliography, following the referencing conventions required for this type of work. I understand that plagiarism and self-plagiarism are treated as academic misconduct under the U.Porto Code of Ethics, and I have acted accordingly throughout.

Generative AI tools were used at certain points of this work, mainly to help polish the written text and to support parts of the code developed for the data analysis. Their role was limited to assisting with these tasks; I reviewed everything they produced, checked it for mistakes or unsupported claims, and took full ownership of the final content, the analyses and the conclusions presented here.

André Ribeiro Oliveira

Contents

Abstract iv

The United Nations Sustainable Development Goals v

Acknowledgements vi

Institutional acknowledgements vii

Declaration of honour viii

1 Introduction 1
- 1.1 Context and motivation . . . 1
- 1.2 Research Objectives . . . 2
- 1.3 Dissertation structure . . . 3

2 Background and state of the art in tool condition monitoring 4
- 2.1 Fundamentals of tool condition monitoring . . . 4
  - 2.1.1 Machining and drilling processes . . . 4
  - 2.1.2 Tool wear and tool life . . . 6
  - 2.1.3 Data acquisition system . . . 8
  - 2.1.4 Tool condition assessment and decision-making methods . . . 11
- 2.2 Systematic literature review . . . 13
  - 2.2.1 Search strategy . . . 13
  - 2.2.2 Results of the search . . . 14
  - 2.2.3 Summary of selected studies . . . 15

3 Experimental Setup and data acquisition 18
- 3.1 Experimental setup . . . 18
  - 3.1.1 Machining center . . . 18
  - 3.1.2 Workpiece material . . . 20
  - 3.1.3 Sensor Instrumentation . . . 20
  - 3.1.4 Data acquisition . . . 22
- 3.2 Design of experiments . . . 24
  - 3.2.1 Solid carbide drill design of experiment . . . 24
  - 3.2.2 High speed steel drill design of experiments . . . 25
  - 3.2.3 G-code programming and machining strategy . . . 26
  - 3.2.4 Tool wear measurement . . . 27
- 3.3 Dataset generation and preparation . . . 29
  - 3.3.1 Dataset: Solid carbide tools . . . 29

3.3.2 Dataset: High speed steel tools . . . 30
3.3.3 Dataset labelling and class definition . . . 31
3.3.4 Dataset structure . . . 33

4 Methodological framework for tool condition classification 35
4.1 Creation of the datasets . . . 35
4.1.1 Data cleaning and segmentation . . . 35
4.1.2 Denoising pipeline . . . 37
4.2 Feature extraction . . . 39
4.2.1 Classical path: Handcrafted features . . . 39
4.2.2 CNN path: Raw windowed signal . . . 40
4.3 Class imbalance handling . . . 41
4.3.1 Undersampling . . . 41
4.3.2 Augmentation . . . 42
4.4 Normalisation . . . 42
4.5 Model architectures . . . 43
4.5.1 Random forest . . . 43
4.5.2 Extreme gradient boosting . . . 43
4.5.3 Multi-Branch 1D-CNN . . . 43
4.6 Validation strategy . . . 45
4.6.1 Run-level cross validation - Split A . . . 45
4.6.2 Hole-level cross validation - Split B . . . 45
4.7 Experimental grid . . . 46
4.8 Evaluation metrics . . . 46
4.9 Hardware and software . . . 47

5 Results and discussion 48
5.1 Overall results . . . 48
5.2 Best-performing configurations . . . 50
5.2.1 Split comparison . . . 56
5.3 Computational performance . . . 57
5.3.1 Split comparison . . . 60
5.4 Per-run performance variance . . . 60
5.4.1 Split comparison . . . 64
5.5 Ablation study . . . 65
5.5.1 Effect of peck selection . . . 65
5.5.2 Effect of channel set . . . 68
5.5.3 Effect of window size . . . 71
5.5.4 Split comparison . . . 72

6 Conclusions and future perspectives 75
6.1 Summary of main findings . . . 75
6.2 Contributions of the present work . . . 76
6.3 Limitations of the study . . . 77
6.4 Future research directions . . . 77

References 79

A G-code used in the experiments 83

List of Figures

2.1 Main machining operations [8]. 5
2.2 Drill geometry parameters [8]. 6
2.3 Types of wear on cutting tools: (a) flank wear; (b) crater wear; (c) notch wear; (d) nose radius wear; (e) comb (thermal) cracks; (f) parallel (mechanical) cracks; (g) built-up edge; (h) gross plastic deformation; (i) edge chipping or frittering; (j) chip hammering; (k) gross fracture [8]. 7
2.4 V_B characterisation in drilling [8]. 8
2.5 Examples of dynamometers for force measurement. 9
2.6 Piezoelectric accelerometer [22]. 10
2.7 Acoustic emission sensor [22]. 10
2.8 Hall effect sensors [23]. 11

3.1 Machining center (*HAAS VF-2*) used. 19
3.2 Side view of the drill bits used in this study: top, solid carbide drill *HBDPCE/W 2 03D 0590-066028*; bottom, high-speed steel drill *DL105060*. 19
3.3 Workpiece before and after being drilled. 20
3.4 Position of the sensors on the MC. 22
3.5 Schematic representation of the data acquisition architecture. 23
3.6 Final full-factorial experimental plan (3 feed levels × 3 speed levels). 26
3.7 Image acquisition setup used for tool wear measurement. 28
3.8 Example microscope images with V_B measurements performed in *ImageJ*. 29
3.9 Wear measurements of a test run of a solid carbide drill to estimate the wear rate. 30
3.10 Wear measurements made in the HSS experiments. 30
3.11 Measured holes per Run. 32
3.12 Distribution of the points per class per Run. 32
3.13 Hierarchical structure of one `.h5` experimental file. One path is fully expanded to show all four levels; all remaining nodes follow the same organisation. 34

4.1 Raw F_z signal (in blue) recorded during a representative drilling sequence with 3 holes, before any filtering. Yellow line shows the digital input state that marks the beginning and end of a hole in the g-code. 37
4.2 Peck boundary detection applied to the smoothed F_z signal. The moving average filter output (solid line in blue), the detection threshold (dashed line in red), and the identified peck start (green circles), end boundaries (red circles) and the digital input state (yellow line) are shown for the representative drilling sequence with 3 holes. 37
4.3 Training set class distribution before and after each imbalance handling strategy, shown for a representative experiment. 41

5.1 Confusion matrix and per-class metrics for XGBoost's best-performing configuration on Split A. 50
5.2 Confusion matrix and per-class metrics for Random Forest's best-performing configuration on Split A. 51
5.3 Confusion matrix and per-class metrics for the CNN's best-performing configuration on Split A. 51
5.4 Training curve for the CNN's best-performing configuration on Split A. 52
5.5 Real V_b vs predicted class in the best configuration of split A. 53
5.6 Confusion matrix and per-class metrics for the CNN's best-performing configuration on Split B. 54
5.7 Training curve for the CNN's best-performing configuration on Split B. 54
5.8 Confusion matrix and per-class metrics for the XGBoost best-performing configuration on Split B. 55
5.9 Confusion matrix and per-class metrics for the random forest best-performing configuration on Split B. 55
5.10 Real V_b vs predicted class in the best configuration of split B. 56
5.11 Training time across all configurations, by model and validation split. 58
5.12 Inference time across all configurations, by model and validation split. 58
5.13 Per-run macro F1 across the six main-grid configurations, Split A. 61
5.14 Per-run macro F1 across the six main-grid configurations, Split B. 63
5.15 Effect of peck selection on Split A macro F1, at each model's best configuration. 66
5.16 Effect of peck selection on Split B macro F1, at each model's best configuration. 68
5.17 Effect of channel set on Split A macro F1, at each model's best configuration. . . 69
5.18 Effect of channel set on Split B macro F1, at each model's best configuration. . . 70
5.19 Effect of window size on Split A macro F1, for the top-3 CNN configurations. . . 71
5.20 Effect of window size on Split B macro F1, for three CNN configurations (1.0 s vs 2.0 s). 72
5.21 Best-worst range for the CNN on each ablation axis, by split. 73

List of Tables

2.1 Summary of selected TCM studies included in the literature review. 14

3.1 Sensors used in the experimental setup. 21
3.2 Experimental plan adopted in this study. 25
3.3 Run to file conversion. 33

4.1 Denoising stages applied per sensor channel. 39
4.2 Handcrafted features computed per sensor channel. 40
4.3 Architecture of each CNN branch. C_g is the number of input channels for the group; width doubles after the first block. 44
4.4 Experimental grid axes and their evaluated values. 46
4.5 Hardware and software configuration. 47

5.1 Experimental grid and classification performance, ordered by macro F1. 49
5.2 Best-performing configuration per model on Split A. 50
5.3 Best-performing configuration per model on Split B. 53
5.4 Split A per-run macro F1 by model, averaged across the six main-grid configurations. 62
5.5 Split B per-run macro F1 by model, averaged across the six main-grid configurations. 64
5.6 Per-run macro-F1 spread and standard deviation, Split A vs Split B. 64
5.7 Effect of peck selection on Split A macro-F1 and accuracy, at each model's best configuration. 66
5.8 Effect of peck selection on Split B macro-F1 and accuracy, at each model's best configuration. 67
5.9 Effect of channel set on Split A macro-F1 and accuracy, at each model's best configuration 69
5.10 Effect of channel set on Split B macro-F1 and accuracy, at each model's best configuration. 70
5.11 Range between the CNN's best and worst results on each ablation axis, by split. . 73

List of Acronyms

AE	Acoustic emission
CNN	Convolutional neural network
CPU	Central processing unit
DAQ	Data acquisition
DoE	Design of experiments
GPU	Graphics processing Unit
HMM	Hidden markov model
HSS	High-speed steel
LOOCV	Leave-One-Out cross-validation
LSTM	Long short-term memory
MC	Machining center
RF	Random forest
RMS	Root mean square
SVM	Support vector machines
TCM	Tool condition monitoring
V_B	Flank wear
XGBoost	Extreme gradient boosting

Chapter 1

Introduction

This chapter introduces the motivation and context for the work developed in this dissertation, defines the objectives pursued, and outlines the structure of the subsequent chapters.

1.1 Context and motivation

Manufacturing relies heavily on machining processes, which play a central role across industries such as aeronautics, energy, aerospace, automotive manufacturing, mould engraving, and fluid mechanics [1, 2]. However, these processes inevitably consume cutting tools, which suffer wear due to friction, heat, and chemical reaction at the tool–workpiece interface. When tool wear progresses beyond acceptable limits, the resulting poor surface quality and dimensional errors can lead to part rejection and increased production costs [3]. In the worst cases, sudden tool breakage can also endanger operator safety and damage the machine itself [4].

The economic effects of tool wear are substantial. Tool failures and wear-related defects contribute to increased downtime (20%) [5], machine stoppages, scrapped parts, and higher operational expenses. According to Zhang and Zhang [6], the cost of cutting tools and their replacement can constitute up to 12% of total production costs, highlighting the need to address and mitigate this challenge.

This work focuses specifically on the drilling process, as it is one of the most widespread machining operations in manufacturing. López de Lacalle et al. [7] reported that drilling accounts for approximately 25% of total machining time in aircraft component fabrication. Its economic significance is further increased by the fact that drilling is typically performed in the final stages of part production [8], meaning that tool failure at this step can result in the scrapping of a fully machined, high-value component.

To address this issue, industry commonly adopts conservative preventive strategies in which cutting tools are replaced based entirely on experience [9]. Although this approach reduces the risk of unexpected failure, it often results in inefficient tool utilisation, with only 50% to 80% of the available tool life being effectively used [10].

To overcome these limitations, Tool Condition Monitoring (TCM) systems have been developed. These systems are generally classified into two main categories: direct and indirect monitoring methods. Direct methods rely on the physical measurement of tool wear using techniques such as optical inspection, electrical resistance measurements, proximity sensors, or machine vision systems. In contrast, indirect methods estimate tool condition using measurable process signals such as force, vibration, acoustic emission, or spindle current that are correlated with tool wear progression. Instead of directly observing wear, these methods analyse sensor outputs to infer the state and degradation level of the cutting tool [11].

1.2 Research Objectives

The main objective of this work is to develop and evaluate a TCM framework for the drilling process, based on indirect sensing methods and data-driven techniques, capable of estimating tool wear and supporting decision-making in machining operations. This work is structured around two main contributions: the construction of a labelled multi-sensor drilling dataset, and the development and comparison of data-driven models for tool wear classification.

The following objectives were defined for the development of the proposed tool condition monitoring framework:

- Design and execute a controlled drilling experimental campaign on AISI 316L stainless steel, covering a range of cutting conditions and acquiring multi-sensor data, including cutting forces, spindle current, vibration, and acoustic signals.
- Measure flank wear periodically to establish the ground truth and develop a structured dataset through the cleaning, segmentation, and labelling of the acquired signals.
- Develop a data-driven tool condition monitoring framework combining signal processing, feature extraction, and machine learning methods for tool wear classification.
- Evaluate and compare the proposed approaches under two validation strategies, assessing their performance under seen cutting conditions and their generalisation to unseen cutting conditions.
- Analyse the influence of the main processing and modelling choices on classification performance and their potential applicability to industrial machining environments.

In addition, this work aims to contribute to a deeper understanding of the relationship between machining signals and tool wear mechanisms, exploring how different sensing modalities capture complementary information about the process.

1.3 Dissertation structure

This dissertation contains five further chapters. Chapter 2 provides the theoretical background and a systematic review of the literature on tool condition monitoring in drilling operations. Chapter 3 describes the experimental setup used to collect the data, including the machining centre, cutting tools, workpiece material, sensor instrumentation, and design of experiments. It also presents the datasets generated during the experimental campaign, covering data cleaning, segmentation, labelling, and storage. Chapter 4 details the methodological framework, including signal preprocessing, feature extraction, class imbalance handling, model architectures, validation strategy, and evaluation metrics. Chapter 5 presents and analyses the experimental results. Finally, Chapter 6 summarises the main findings, presents the contributions, discusses the limitations of the work, and proposes directions for future research.

Chapter 2

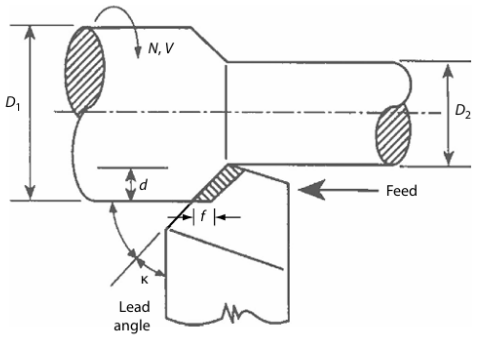
Background and state of the art in tool condition monitoring

This chapter provides the theoretical and bibliographic foundation for the work developed in subsequent chapters. It is organised into two main sections. The first presents the theoretical background covering the fundamental concepts relevant to this study, including the machining process, tool wear mechanisms, sensor modalities used for indirect monitoring, and the decision-making methods commonly applied in TCM systems. The second section presents a systematic literature review of recent works on TCM in drilling operations, analysing the methodological trends and identifying the gaps that motivate the present work.

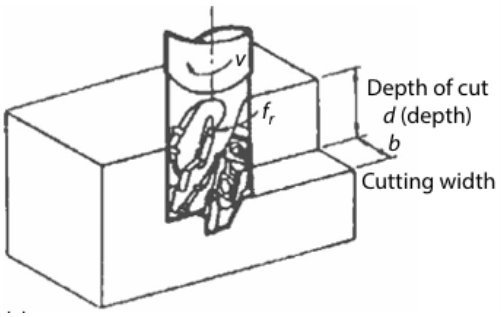
2.1 Fundamentals of tool condition monitoring

2.1.1 Machining and drilling processes

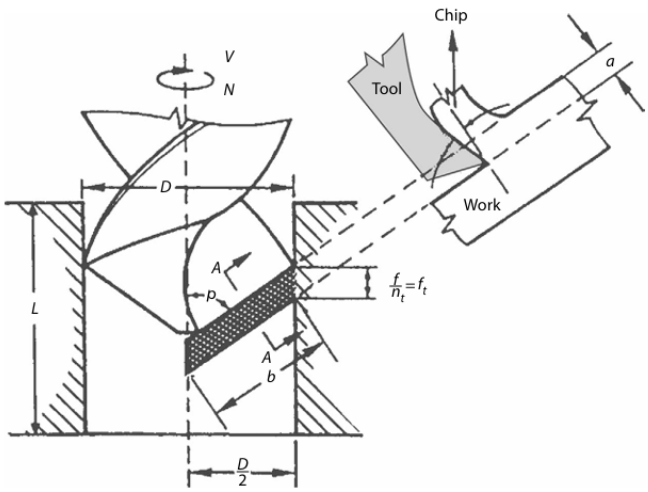
Machining is a subtractive manufacturing process in which material removal is performed using cutting tools to achieve precise dimensions, surface finish, and geometric features on a workpiece. The three most common operations are turning, milling, and drilling, represented in Figure 2.1. Turning produces cylindrical parts by rotating the workpiece against a stationary cutting tool. Milling removes material using a rotating multi-point cutter to create flat or complex surfaces. Drilling produces round holes using a rotating drill bit [12, 13].



(a) Turning



(b) Milling



(c) Drilling

Figure 2.1: Main machining operations [8].

As this work focuses on the drilling process, it is essential to understand the fundamental aspects that govern its behaviour. This begins with the key process parameters, namely feed rate and spindle speed. The feed rate, typically expressed in mm/rev, defines the drill's axial advancement per revolution and directly influences chip load, cutting forces, and tool wear. Spindle speed, expressed in revolutions per minute (rpm), determines the cutting speed at the tool–workpiece interface and significantly affects temperature generation, tool life, and surface finish.

In addition to process parameters, it is important to consider the forces involved in drilling. The operation is primarily characterised by thrust force, acting along the tool's axial direction, and torque, resulting from rotational motion. These forces are generated by the interaction between the cutting edges and the material, and are strongly influenced by cutting conditions, material properties, and tool geometry. Excessive forces can increase tool wear, cause dimensional inaccuracies, and lead to tool failure [14].

Another critical factor is drill-bit geometry. Key geometric features include the point angle,

helix angle, lip geometry, and chisel edge. The point angle affects the distribution of cutting forces and penetration capability, the helix angle influences chip evacuation and cutting efficiency, and the chisel edge contributes to thrust force generation. Variations in these geometric parameters significantly impact cutting performance, force magnitude, heat generation, and ultimately tool wear and hole quality.

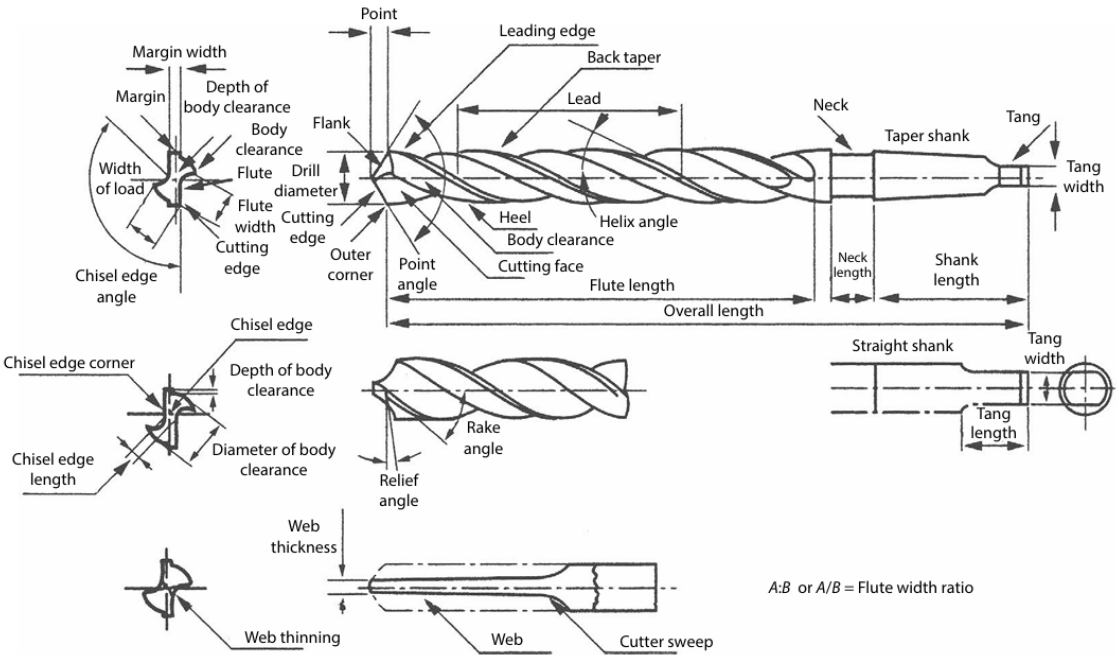


Figure 2.2: Drill geometry parameters [8].

As the tool wears down, the increased contact between the tool's relief surface and the hole wall raises ploughing forces, causing thrust force and torque to rise monotonically with tool degradation [15, 16]. Tracking sensor signals and connecting their changes over time to these force signals or other mechanisms that may result from wear accumulation across successive operations provides the theoretical basis for the TCM system this work seeks to establish.

2.1.2 Tool wear and tool life

In machining operations, cutting tools experience several distinct wear types that alter their geometry and affect performance. As illustrated in Figure 2.3, the most significant types include crater wear, notch wear, and flank wear (V_B).

Crater wear is characterised by the formation of a visible depression on the rake face of the cutting tool, typically occurring in regions of high friction and elevated temperature. Notch wear manifests as a pronounced groove at the point where the cutting edge reaches the maximum depth of cut, and it is particularly common when machining materials with hard surface layers or high-alloy steels. Flank wear appears as a relatively uniform wear land on the clearance face, resulting primarily from abrasive and adhesive interactions between the tool and the workpiece [17, 18]

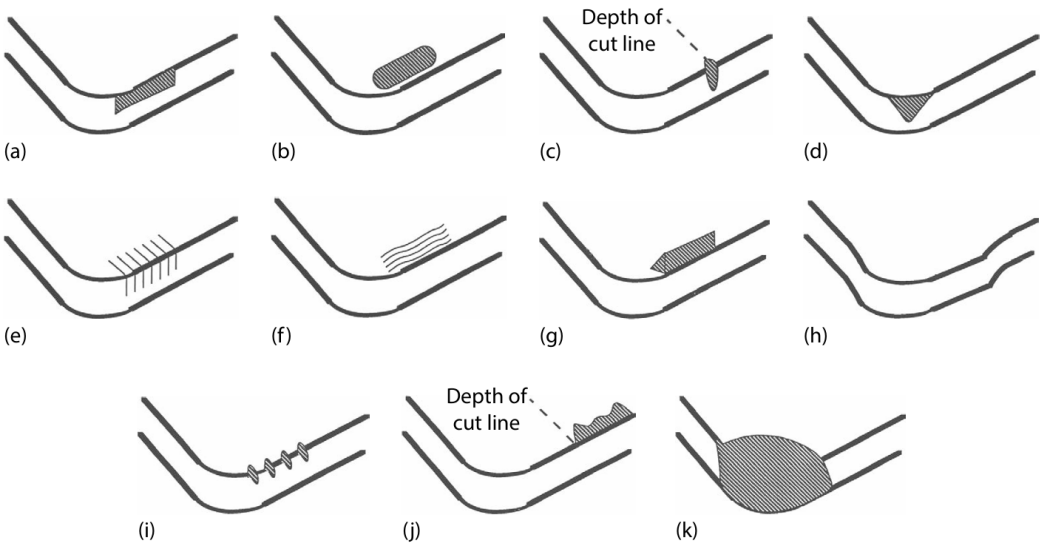
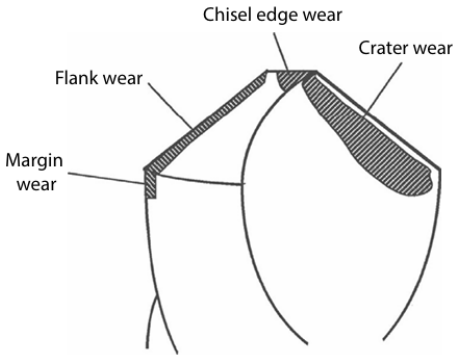


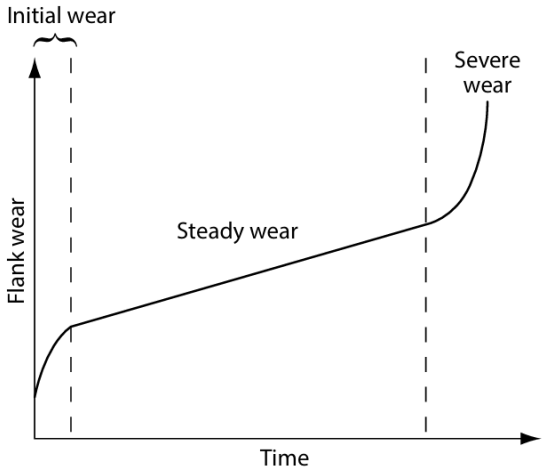
Figure 2.3: Types of wear on cutting tools: (a) flank wear; (b) crater wear; (c) notch wear; (d) nose radius wear; (e) comb (thermal) cracks; (f) parallel (mechanical) cracks; (g) built-up edge; (h) gross plastic deformation; (i) edge chipping or frittering; (j) chip hammering; (k) gross fracture [8].

This work focuses on flank wear because it directly affects machined product quality [18], is the primary factor influencing tool life [19], and is easier to measure and quantify than other wear types [20].

According to ISO 8688-1 [21], tool life is determined by the average V_B width. Under uniform wear conditions, the tool life limit is defined as $V_B = 0.3$ mm. Figure 2.4 shows the characteristic wear patterns for conventional twist drills and the typical flank wear evolution curve. Three distinct regions can be observed: an initial wear region, a steady wear region, and a severe wear region, with the V_B value increasing non-linearly with cutting time throughout all three stages.



(a) Characteristic wear patterns for conventional twist drills.



(b) Evolution of V_B as a function of time.

Figure 2.4: V_B characterisation in drilling [8].

It is worth noting that tool breakage is a distinct failure mode with a stochastic nature. Early breakage may occur at the beginning of the cutting operation or shortly after, and is most likely when machining high-hardness materials with brittle tools, such as ceramic or carbide tools, or under intermittent cutting conditions [4]. Furthermore, with the continuous accumulation of wear, sudden tool breakage may also occur after the severe wear stage.

2.1.3 Data acquisition system

For the purpose of implementing a TCM using indirect methods, the use of sensors is, in most cases, imperative. According to the literature, the most commonly employed sensors in this field include dynamometers for measuring cutting forces, accelerometers for vibration monitoring, acoustic emission (AE) sensors for capturing high-frequency stress waves, Hall-effect sensors for monitoring electrical current, microphones for sound analysis, and thermocouples, pyrometers, or infrared cameras for measuring cutting temperature [4, 16]. Recent literature highlights a clear shift from single-sensor approaches toward multi-sensor systems in TCM [4]. This trend is driven by sensor fusion, which combines data from multiple sources to capture complementary information about the machining process. While individual sensors may be sensitive to specific phenomena, their integration improves robustness, reduces uncertainty, and enhances the accuracy of tool wear estimation. Consequently, a thorough understanding of the different sensor types and the characteristics of the data they generate is essential for designing effective monitoring systems.

To better understand their role in tool condition monitoring, it is important to analyse how each sensor type operates, what physical phenomena it captures, and how these measurements relate to tool wear progression.

Dynamometers measure cutting forces based on the deformation of sensing elements, typically using piezoelectric or strain gauge technology. In drilling operations, they provide force compo-

nents along different axes (Fx, Fy and Fz). Tool wear directly affects cutting mechanics, increasing friction and reducing cutting efficiency, which manifests as a gradual rise in cutting forces. Therefore, force signals are widely regarded as one of the most reliable indicators of tool wear.

Figure 2.5 presents two examples of dynamometers. The first is a table dynamometer, similar to the one used in this work. Although this solution allows measurement of the forces generated during the drilling process, it requires mounting the workpiece on top of the dynamometer. Consequently, the hole position relative to the dynamometer may influence the force measurement and limit the flexibility of the experimental setup. The second example is a wireless rotating dynamometer, which would be preferable for this application because it has less impact on the existing setup and greater flexibility in hole positioning. Because the dynamometer is mounted directly on the spindle and rotates with the tool, the force-measurement point remains aligned with the drilling location, ensuring that the measured forces depend less on the hole position within the workpiece.



(a) Table dynamometer [22]



(b) Wireless rotating dynamometer [22]

Figure 2.5: Examples of dynamometers for force measurement.

Accelerometers measure vibration through inertial effects, commonly using piezoelectric elements that generate an electrical signal proportional to acceleration. As tool wear progresses, the cutting process becomes less stable, often increasing vibration levels and triggering phenomena such as chatter. Changes in vibration amplitude and frequency content can thus provide valuable information about tool degradation and process instability.

Figure 2.6 shows a piezoelectric accelerometer commonly used for vibration measurement in machining applications.



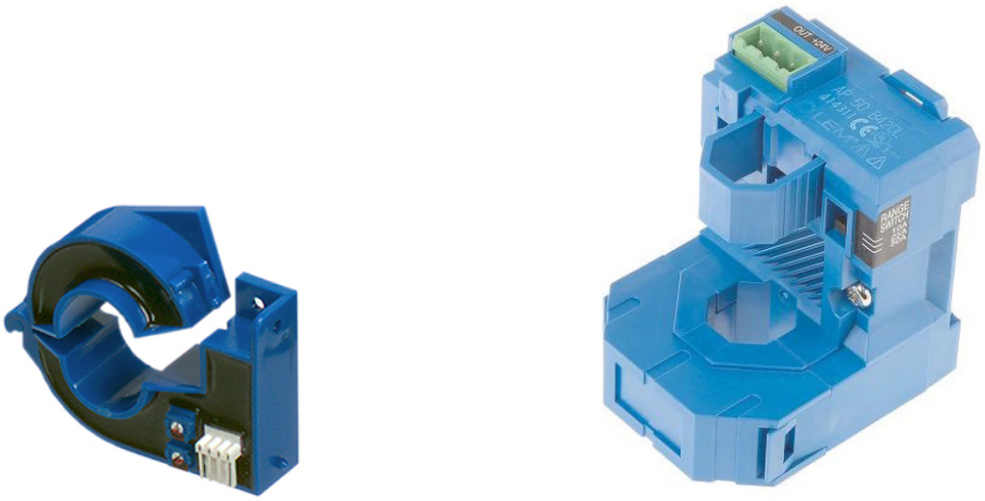
Figure 2.6: Piezoelectric accelerometer [22].

AE sensors capture high-frequency elastic waves generated by rapid energy releases within the material, such as micro-fractures, plastic deformation, and friction at the tool-workpiece interface. Unlike conventional vibration sensors, AE sensors detect microscopic events, making them particularly useful for early detection of tool wear and crack initiation. Figure 2.7 shows an example of a piezoelectric acoustic emission sensor.



Figure 2.7: Acoustic emission sensor [22].

Hall-effect sensors measure electrical current by detecting the magnetic field generated by current flow in the spindle motor. Since cutting forces increase with tool wear, the mechanical load on the spindle also rises, leading to higher current consumption. Although less direct than force measurements, spindle current provides a simple and cost-effective proxy for monitoring tool condition, especially in industrial environments where additional sensors may not be feasible. Examples of Hall-effect current sensors are shown in Figure 2.8.



(a) Instantaneous current (b) RMS current

Figure 2.8: Hall effect sensors [23].

Microphones capture airborne sound generated during machining. Variations in cutting conditions, such as tool wear or chip formation irregularities, can alter the process's acoustic signature. Although these signals are more susceptible to environmental noise, advanced signal-processing techniques can extract meaningful features related to tool condition.

Temperature sensors, such as thermocouples or infrared devices, measure the thermal conditions at or near the cutting zone. As tool wear increases, friction between the tool and workpiece also rises, leading to higher temperatures. Elevated temperatures not only accelerate further wear but also affect surface integrity, making thermal measurements an important complementary indicator.

2.1.4 Tool condition assessment and decision-making methods

To analyse sensor signals and convert them into meaningful information regarding the tool's condition, decision-making algorithms are required. These algorithms establish the relationship between the measured process variables and the wear state of the cutting tool. In the literature, these approaches can be broadly classified into threshold-based, model-based, traditional machine learning methods or deep learning methods.

Threshold-based methods are based on the comparison of signal features with predefined limit values [15]. Although simple and computationally efficient, their main limitation is limited adaptability to changing cutting conditions, which restricts their use in complex and highly dynamic machining processes.

Model-based methods rely on analytical or empirical mathematical models that describe the relationship between cutting parameters and tool wear progression [8]. Unlike threshold-based approaches, these methods rely on physical principles or experimentally fitted equations to estimate tool life and predict wear evolution under specific cutting conditions.One of the most widely

adopted formulations is the Taylor tool life equation [24] and its extended version [25], which relates tool life to cutting speed through an empirical power-law relationship. Over the years, researchers have developed more advanced model-based approaches to improve predictions of tool wear and tool life under realistic machining conditions. These models are typically formulated as functions of cutting speed, feed rate, and depth of cut, while explicitly incorporating the physical mechanisms governing tool degradation. In particular, they consider mechanical loading, frictional effects, plastic deformation, and thermal phenomena at the tool–chip interface. By integrating cutting mechanics and heat-generation principles, these physics-based models aim to provide more reliable wear estimates within controlled operating domains [26].

TCM problems can be formulated as either classification or regression tasks. Classification assigns the tool to predefined wear states, whereas regression predicts a continuous wear measure, such as flank wear V_B. In this work, a classification approach was adopted because the available dataset is relatively limited and the objective is to distinguish between discrete tool condition states. Compared with directly predicting continuous wear values, this formulation can provide a more sample-efficient learning problem for the available data.

Traditional machine learning methods are among the most widely used approaches in TCM because they perform well with limited datasets and have relatively low computational cost. Random Forest (RF) was introduced by *Leo Breiman* [27] and is an ensemble learning technique that constructs multiple decision trees using different subsets of the data and input features. The final prediction combines the outputs of all trees, improving generalisation and reducing overfitting. In machining applications, RF has been successfully used for multisensor tool wear monitoring, showing a strong ability to predict tool wear [28, 29]. Support Vector Machines, originally proposed by *Cortes and Vapnik* [30], are supervised learning models that determine an optimal decision boundary by maximising the margin between data classes. By employing kernel functions, SVM can efficiently model nonlinear relationships between sensor signals and tool wear. This technique has demonstrated high accuracy in tool condition monitoring applications using both single-sensor and multi-sensor approaches, providing robust performance even in high-dimensional feature spaces and under noisy machining conditions [31, 32]. Hidden Markov-based models (HMM), such as those presented in [33], have been widely applied to represent the progressive and stochastic nature of tool wear. In these approaches, the actual wear condition is considered a hidden state, while the observable variables correspond to features extracted from sensor signals. An extension of the standard HMM is the Hidden Semi-Markov Model, in which each hidden state generates a sequence of observations instead of a single observation. This allows the explicit modelling of the state duration, introducing a temporal dependency that is particularly suitable for representing the degradation process of industrial components [34].

In recent years, deep learning techniques have gained significant attention in TCM because they can automatically learn representative features directly from raw or minimally processed sensor signals, eliminating the need for manual feature engineering [2]. This characteristic is particularly advantageous in machining environments, where the relationship between process signals and tool degradation is highly non-linear and complex. Among deep learning architec-

tures, Convolutional Neural Networks (CNN) have become one of the most widely adopted approaches. CNN are specifically designed to extract hierarchical feature representations through convolutional operations, enabling the identification of local patterns and invariant characteristics in the input data [35]. In TCM applications, CNN are frequently applied to time–frequency signal representations, such as spectrograms, scalograms, and wavelet transform maps, where they have demonstrated excellent performance in tool wear classification [36, 37]. Beyond stand-alone implementations, CNN are often integrated with other neural network architectures to further improve performance. In such hybrid frameworks, CNN typically serves as an automatic feature extractor and dimensionality-reduction module, transforming high-dimensional raw data into compact, discriminative feature representations [9]. Another important family of deep learning models is Recurrent Neural Networks (which derive from the model proposed by Elman in [38], particularly Long Short-Term Memory (LSTM) networks. These architectures are designed for sequential data modelling, as they incorporate internal memory mechanisms that can learn long-term temporal dependencies [39]. Because tool wear is inherently time-dependent and progressive, LSTM networks are well suited to model the temporal evolution of machining signals and predict future tool-wear states or the remaining useful life of the cutting tool. Their ability to process time-series data directly from sensors such as cutting force, vibration, or acoustic emission has led to highly accurate systems [40, 41].

2.2 Systematic literature review

This section presents a structured review of the scientific literature on tool condition monitoring in drilling operations. The goal is to map the current state of the art, identify the most relevant methodological approaches, and establish the foundation for the decisions made in the subsequent chapters.

2.2.1 Search strategy

To ensure broad coverage, the literature search was conducted across five scientific databases: Scopus, Web of Science, IEEE Xplore, ScienceDirect, and Google Scholar. The following keywords and their combinations, were used to guide the search: *drilling*, *tool wear*, *tool condition monitoring*, and *multisensor fusion*.

The search was restricted to publications from 2020 to 2026, ensuring recency while maintaining a manageable scope. No restriction was applied regarding journal type; however, only peer-reviewed sources were considered.

Following the initial retrieval, a set of inclusion and exclusion criteria was applied to filter the results:

Inclusion criteria:

- Studies involving hard to machine alloys or steels.

Exclusion criteria:

- Studies limited exclusively to tool breakage detection, with no predictive or monitoring component
- Purely analytical or finite element method studies containing no experimental sensor data
- Non-English publications

2.2.2 Results of the search

Table 2.1 summarises the key characteristics of the studies included in this review. For each paper, the workpiece material, sensor modalities, data processing approach, decision-making method, and monitoring objective are listed.

Table 2.1: Summary of selected TCM studies included in the literature review.

Paper	Material	Operation	Sensors	Data Processing	Model	Objective
[37]	1018 steel	Milling	Microphone (acoustic)	STFT spectrogram	2D CNN	Wear state classification
[9]	TC4	Milling	Accelerometer (3-axis), current (Hall effect)	Time/freq. features, wavelet energy	CNN-SBULSTM	RUL prediction
[40]	Stainless steel	Milling	Force (3-axis), vibration (3-axis), AE	Statistical features, process info fusion	Stacked LSTM	Tool wear prediction
[42]	Stainless steel	Milling	Force (3-axis), vibration (3-axis), AE	Time/freq./time-freq. features, correlation analysis, PCA	SSAE BPNN	Tool wear prediction
[41]	Stainless steel J45	Face milling	AE, vibration	Morlet wavelet scalogram, SinGAN augmentation	Stacked LSTM	Tool wear prediction
[43]	Stainless steel	Milling	Multi-sensor	Spatial-temporal graph construction	Gated GCN	RUL prediction
[44]	Ni-based superalloy	Drilling	Force	SSA noise removal, PCA, one-hot encoding	SSA-BLSTM	Wear type detection
[45]	Inconel 718	Drilling	Force, vibration, AE	Multi-sensor fusion, data augmentation	CNN-LSTM	Tool wear prediction

Continued on next page

Table 2.1 – *continued from previous page*

Paper	Material	Operation	Sensors	Data Processing	Model	Objective
[46]	AISI 1045	Drilling	AE (piezoelectric)	WPD, time/freq./time-freq. features, Min-Max norm.	k-NN family (WKNN, LWL, IBk)	Wear state classification
[47]	AISI 1045	Drilling	AE (piezoelectric)	HHT, EMD, band-pass filtering, norm.	CatBoost	Wear state classification
[48]	AISI 9840	Drilling	Vibration, AE, power, torque	Statistical features	NN, Random Forest	Wear state classification

2.2.3 Summary of selected studies

The studies reviewed reveal consistent patterns in how tool condition monitoring is approached across the literature, both in terms of sensor selection and decision-making methodology.

Sensor Modalities

Across the reviewed works, three sensor types dominate indirect TCM: accelerometers, AE sensors, and force or torque dynamometers. Vibration signals captured by accelerometers are the most widely used modality, owing to their low intrusiveness and ease of integration into existing machine tool setups [9, 44, 48]. AE sensors appear frequently in drilling-specific studies, where their sensitivity to high-frequency transient events associated with chip formation and flank contact makes them particularly well-suited for detecting early-stage wear [46–48]. Force sensors, while providing rich and directly wear-related information, are less common in recent works due to their higher cost and the mechanical complexity of installation [49]. Spindle current and power signals are also employed as indirect proxies for cutting load, offering a completely non-intrusive alternative that can be acquired directly from the machine controller [9]. A growing number of studies adopt multi-sensor fusion strategies, combining two or more of the above modalities to capture complementary information and improve monitoring robustness [40, 42, 45].

Signal processing and feature extraction

Regardless of the sensor used, the raw signals are rarely fed directly into the decision-making model. The vast majority of reviewed works apply a structured signal processing pipeline prior to model training. In the time domain, statistical descriptors such as root mean square (RMS), kurtosis, skewness, peak-to-peak amplitude, and standard deviation are widely extracted, as these statistics capture changes in signal energy and impulsiveness associated with tool degradation [46, 47]. In the frequency domain, the Fast Fourier Transform (FFT) is routinely applied to identify dominant frequency components and spectral shifts induced by wear progression [46]. Time-frequency

methods are increasingly preferred for their ability to represent non-stationary signal behaviour. Wavelet Transform (WT) and Wavelet Packet Decomposition (WPD) are the most frequently used in this category, allowing sub-band energy decomposition across wear-sensitive frequency ranges [41, 46]. More recent works employ the Hilbert-Huang Transform (HHT), which performs adaptive decomposition via Empirical Mode Decomposition (EMD) and is particularly effective on non-stationary, nonlinear signals such as those produced during drilling [47]. Dimensionality reduction techniques, notably Principal Component Analysis (PCA), are applied downstream of feature extraction to eliminate redundancy and reduce overfitting risk, especially in multi-sensor configurations with large feature vectors [42, 44].

Decision making models

The reviewed studies span a wide spectrum of modelling approaches, from classical machine learning to advanced deep learning architectures. Among classical methods, Support Vector Machines (SVM), k-Nearest Neighbour classifiers (k-NN) and their weighted variants, Random Forest, and gradient boosting models such as CatBoost appear frequently in classification tasks [46–48]. These methods are generally compact, computationally efficient, and perform well when the feature engineering step is well-designed. However, their performance depends heavily on expert-driven feature selection, which limits scalability and adaptability [49, 50].

Deep learning models have gained significant traction in more recent publications [49, 50]. LSTMs are particularly prominent, as their ability to capture temporal dependencies in sequential sensor data aligns well with the time-series nature of machining signals [40, 41]. CNNs appear both as standalone classifiers often applied to spectrograms or time-frequency images derived from the raw signals [37] and as feature extractors in hybrid architectures. The combination of CNN and LSTM into hybrid models has emerged as a recurrent strategy, where the convolutional layers extract local spatial features from the signal representation and the LSTM layers model the temporal evolution of tool degradation [9, 45]. Bidirectional variants (BiLSTM) further extend this by processing the sequence in both forward and backward directions, which has shown particular effectiveness in drilling wear type detection [44]. Generative approaches such as Singular Generative Adversarial Networks (SinGAN) have also been explored to address the problem of limited training data by synthesising additional wear samples from small experimental datasets [41].

General trends and limitations

The overarching trend emerging from the reviewed literature is a clear shift away from classical machine learning pipelines that rely on handcrafted features toward end-to-end deep learning architectures capable of learning representations directly from raw or lightly processed signals [4, 49, 50]. Reviewed works confirm this trajectory and note that hybrid models combining spatial and temporal feature learning consistently outperform single-architecture approaches [50]. Multi-sensor fusion is increasingly treated as a standard design choice rather than an optional extension, given the complementary information provided by different physical channels [42, 43, 49].

Despite this progress, several limitations appear consistently across the literature. The scarcity of labelled experimental data for wear-to-failure studies is a major constraint, as running tools to complete failure under controlled conditions is time-consuming and expensive [41, 50]. Most studies are conducted under fixed or narrowly varied cutting parameters, which limits generalisability to real production environments where speed, feed, and depth of cut change continuously [4, 49]. Standardised public datasets are scarce in drilling, in contrast to milling, where benchmarks such as the 2010 PHM Data Challenge are widely adopted [40, 42], making direct method comparison difficult. Finally, the gap between laboratory-validated models and industrial deployment remains largely unaddressed, with few works demonstrating real-time integration into production systems [48].

Chapter 3

Experimental Setup and data acquisition

The development of a reliable TCM system requires the acquisition of high-quality experimental data under controlled and repeatable machining conditions. Chapter 3 presents the complete methodology adopted to generate the dataset used in this work, including the machining setup, cutting tools, workpiece material, sensor instrumentation, data acquisition system, experimental design and dataset generation and preparation. The goal is to clearly and comprehensively describe the conditions under which the drilling tests were performed to ensure transparency and reproducibility.

3.1 Experimental setup

The experimental setup was designed to perform controlled drilling operations while capturing multiple sensor signals associated with the cutting process. The following subsections describe the Machining center (MC), cutting tools, workpiece material, and sensor instrumentation used in the tests. Together, these elements define the physical environment in which the dataset was generated and establish the baseline conditions for the analysis carried out in this study.

3.1.1 Machining center

The experiments were carried out on a MC, specifically a *HAAS VF-2* displayed in Fig 3.1. The *HAAS VF-2* is a three-axis vertical MC equipped with a high-speed spindle capable of reaching 8,000–10,000 rpm, allowing a broad range of machining operations, including milling, drilling, and tapping [51].



Figure 3.1: Machining center (*HAAS VF-2*) used.



Figure 3.2: Side view of the drill bits used in this study: top, solid carbide drill *HBDPCE/W 2 03D 0590-066028*; bottom, high-speed steel drill *DL105060*.

Figure 3.2 shows that two different drills were used during the experimental campaign. The initial tests were performed with the *HBDPCE/W 2 03D 0590-066028*, manufactured by *PALBIT*. This tool is a solid carbide drill with a 5.9 mm diameter, two flutes, a 140° point angle, and internal coolant channels. Its geometric and material characteristics make it suitable for high-performance drilling in stainless steels. The manufacturer's catalogue provides detailed specifications for this tool [52].

However, after the first drilling sequence, it became evident that the solid carbide tool exhibited very low wear progression. Due to limitations in available machining time and workpiece material, it was not feasible to continue the full wear-progression study using this tool. For this reason, the remaining experiments were carried out using a second drill model, the *DL105060* from *YG-1*,

manufactured from high-speed steel (HSS). This tool has a diameter of 6 mm, two flutes, a 135° point angle, and is advertised as being capable of cutting stainless steel. The manufacturer's site provides more details on this tool [53].

3.1.2 Workpiece material

This study used AISI 316L stainless steel, an austenitic alloy widely used in marine, chemical, and offshore applications for its excellent corrosion resistance, good mechanical properties, and low carbon content, which enhances weldability. Despite these advantages, AISI 316L is known for its poor machinability, mainly due to its tendency to work-harden, generate high cutting forces and elevated cutting temperatures [54].

The specimens were manufactured as cylindrical blocks with a diameter of approximately 150 mm and a height of around 30 mm. Each cylinder included an integrated rectangular base section to facilitate rigid clamping in a machine vice, ensuring stable and repeatable positioning during drilling. Figure 3.3 shows the workpiece before and after an experiment.

Before the drilling tests, the top surface of each workpiece was face-milled to obtain a flat and uniform plane. This preparation step ensured consistent conditions between the drill and the material across workpieces by reducing geometric variability.

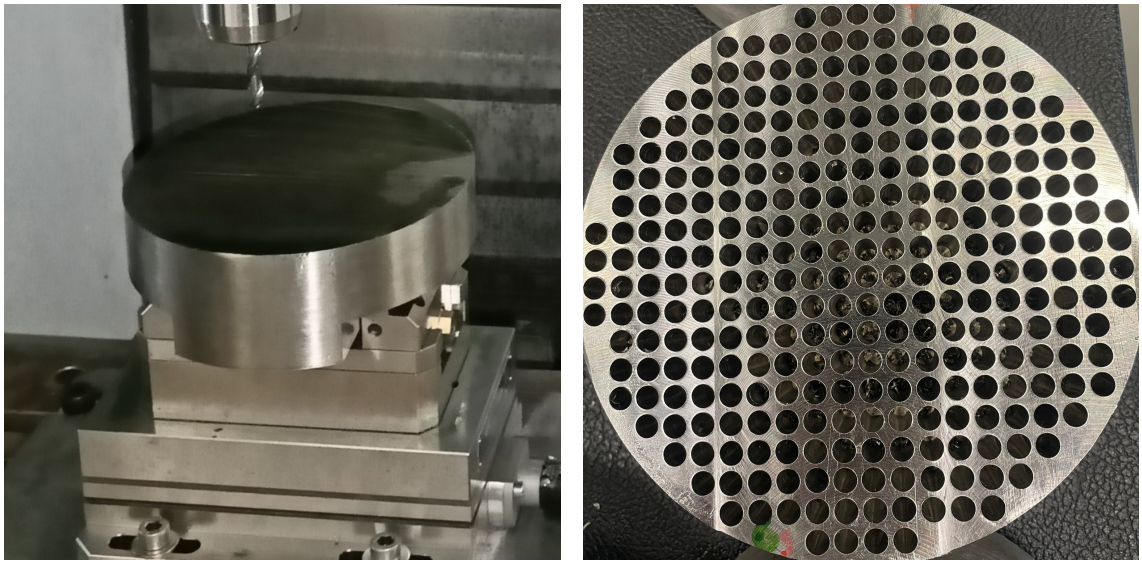


Figure 3.3: Workpiece before and after being drilled.

3.1.3 Sensor Instrumentation

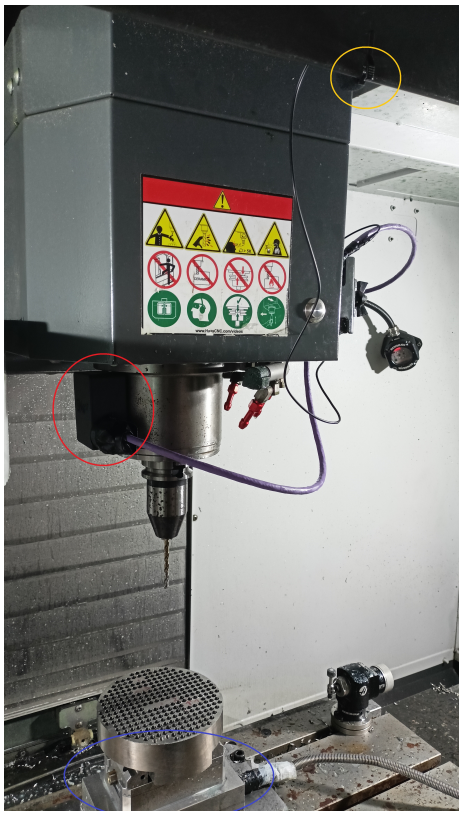
To obtain the sensor data used in this study, multiple measurement devices were integrated into the machining setup, each capturing a different physical phenomenon associated with the drilling process. The combination of force, current, acoustic, and vibration signals provides a multi-modal representation of the cutting conditions.

A table dynamometer from *Kistler* was mounted between the machine table and the workpiece to measure the cutting forces along the three orthogonal directions (F_x, F_y, F_z). A common clip-on microphone was placed inside the machining enclosure, positioned close to the drilling zone, to record the airborne acoustic emissions generated during cutting.

To monitor spindle motor behaviour, two current sensors from [23] were used: one measuring the instantaneous current in a single-phase HTR 50-SB, and another AP 50 B10 measuring the root-mean-square (RMS) current in a different phase. Additionally, a three-axis accelerometer developed by *PALBIT* was mounted on the spindle housing using a magnetic base to ensure rigid coupling. Table 3.1 summarises the sensors used in the experiments, while Figure 3.4 illustrates their placement on the machine.

Table 3.1: Sensors used in the experimental setup.

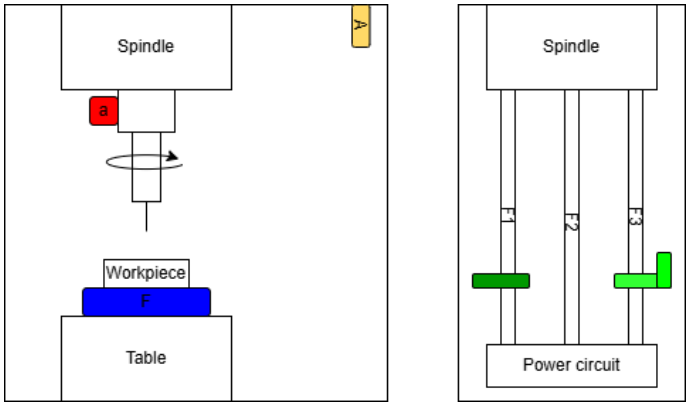
Type of Sensor	Model of Sensor	Signal(s)	Sampling frequency
Dynamometer	9257B [22]	F_x, F_y, F_z	5 kHz
Current Sensor	HTR 50-SB [23]	C_1	5 kHz
Current Sensor RMS	AP 50 B10 [23]	C_{RMS}	5 kHz
Microphone	Clip-On	Audio	48 kHz
Accelerometer	IMU	a_x, a_y, a_z	26.66 kHz



(a) Front view



(b) Electrical panel view



(c) Diagram of the position of the sensors

Figure 3.4: Position of the sensors on the MC.

3.1.4 Data acquisition

The overall data flow used in this study is illustrated in Figure 3.5. The microphone signal was transmitted directly to the computer via USB, while the three-axis accelerometer communicated through an Ethernet interface. The data acquisition (DAQ) system acquired the force, current, and digital trigger (DI) signals and communicated with the computer via USB. The dynamometer

output was first passed through an external signal amplifier (Model 5070A from *Kistler* [22]) to ensure proper scaling and conditioning before entering the DAQ device.

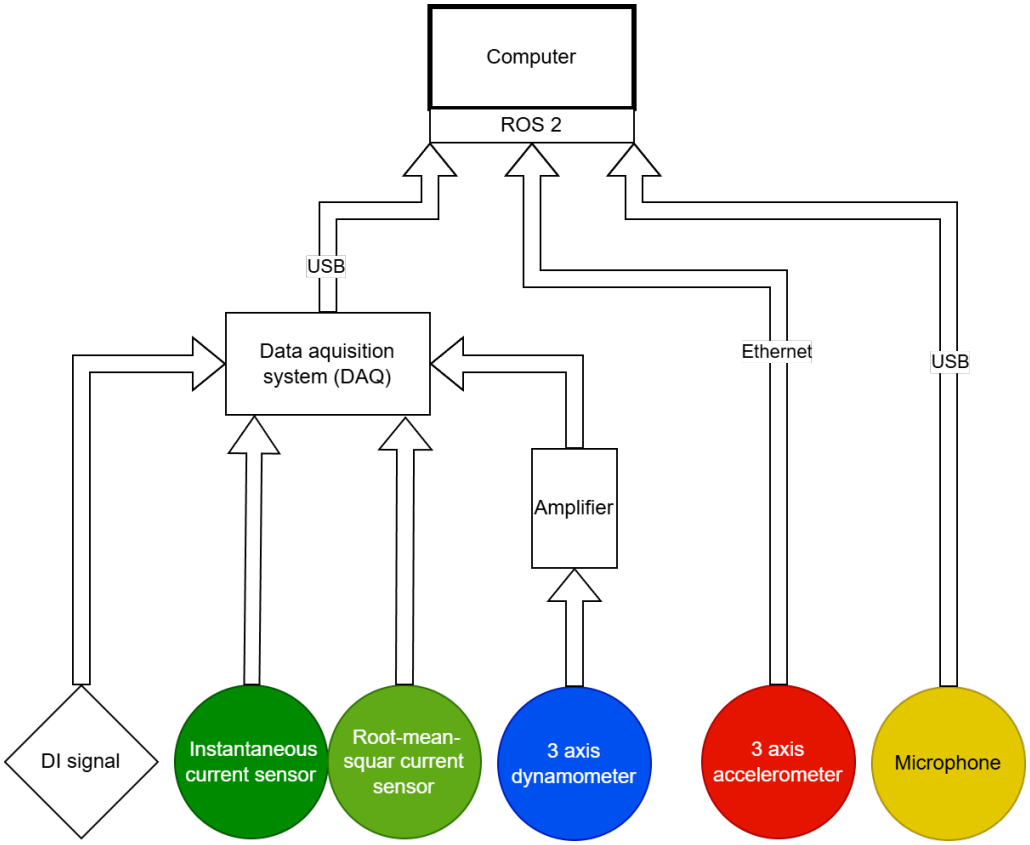


Figure 3.5: Schematic representation of the data acquisition architecture.

The force and current signals were recorded using an *Advantech USB-4704* DAQ system, which supports simultaneous multi-channel acquisition at high sampling rates. The device specifications are in [55]. The DAQ was configured to acquire all channels synchronously, ensuring proper temporal alignment between force, current, and the digital trigger signal.

To guarantee consistent segmentation of the drilling operations, a digital input (DI) trigger was used. This signal was activated at the beginning of each drilling cycle and deactivated at the end, allowing it to serve as a temporal reference for aligning the multi-rate signals during post-processing. The DI signal was recorded through the DAQ alongside the force and current channels.

Data structure

The data were organised into a structured format to facilitate analysis and reproducibility. Drilling operations were grouped in blocks of 10 holes, corresponding to the experimental procedure. For each block, a dedicated folder was created containing:

- **daq.h5**: force, current signals and DI

- **audio.h5**: microphone recordings
- **accel.h5**: three-axis accelerometer data

Each file stores the raw signals together with timestamps, enabling precise reconstruction of the drilling sequence.

3.2 Design of experiments

The design of experiments (DoE) was designed to systematically study the influence of machining parameters on sensor signals and the progression of tool wear. The objective was to obtain a structured and representative dataset while ensuring that the total machining time, tool life, and material availability remained compatible with the experimental campaign's practical constraints. All experiments were planned to continue until the flank wear reached $V_B = 0.3$ mm, which is the commonly accepted end-of-life criterion for drilling tools according to the ISO standard 8688-1

The experimental planning process evolved in two stages. Initially, an approach for the solid carbide drills was created. However, after preliminary tests and a significant number of holes in the first run, this strategy was abandoned due to the change in drill type and replaced with a full-factorial grid-search DoE. This subsection describes both approaches and the reasons for the transition.

3.2.1 Solid carbide drill design of experiment

Two machining parameters were selected as the control variables for the experimental campaign:

- Spindle speed (n)
- Feed rate (f)

All remaining parameters, such as depth of cut, tool geometry, cooling conditions, and material properties, were kept as constant as possible throughout the experiments to isolate the effect of the selected variables.

Each control variable was set at three distinct levels, corresponding to typical operating conditions within the selected machining range as specified by the manufacturer's recommended values.

Instead of performing all possible combinations of the factors, the experimental plan was structured into three independent runs. In each run, the feed rate was held constant, while the spindle speed was varied systematically during machining.

More specifically, for each run:

- A fixed feed rate level was selected;
- Drilling operations were performed sequentially;

- The spindle speed was changed every 10 holes, cycling through the three predefined speed levels.

To monitor tool wear progression, images of the cutting tool were acquired periodically throughout the experiments. Specifically, one image of the tool was captured after every 10 holes.

This strategy was adopted not only to reduce the total number of experiments, but also based on findings reported in the literature, which indicate that feed rate has a more significant influence on tool wear progression than spindle speed [56]. Therefore, maintaining a constant feed rate within each run allows a clearer assessment of its effect, while still enabling the evaluation of spindle speed variations.

Additionally, varying cutting speed within the same run minimises variability associated with tool changes and setup conditions.

The complete experimental plan is summarised in Table 3.2.

Table 3.2: Experimental plan adopted in this study.

Run	Feed Rate (f)	Cutting speed sequence (m/min)
1	0.16	$40 \rightarrow 60 \rightarrow 80$
2	0.12	$40 \rightarrow 60 \rightarrow 80$
3	0.08	$40 \rightarrow 60 \rightarrow 80$

Although this approach does not allow full interaction analysis between variables, it is sufficient for identifying trends and for training data-driven models.

However, during the initial drilling trials with the solid carbide tool, it became evident that the tool exhibited extremely slow wear progression. Even after a large number of drilled holes (almost 1400), the measured wear remained minimal. As a result, completing the DoE plan would have required significantly more machining time and workpiece material than available. The solution was to replace the tool with a High-speed steel (HSS) drill to allow measurable wear progression within the project constraints.

3.2.2 High speed steel drill design of experiments

After switching to the HSS drill, a new DoE was defined following a full-factorial grid-search methodology. The same two machining parameters were selected as control variables:

- Spindle speed (n)
- Feed rate (f)

Each parameter was defined at three levels, obtained by applying a variation of +25% and -25% around the nominal cutting conditions recommended by the tool manufacturer in [53]. This resulted in:

$$3 \times 3 = 9 \text{ experimental runs}$$

All other machining conditions, such as tool geometry, cooling, depth of cut, and material were kept constant as much as possible to isolate the effects of the selected parameters.

For each run, a unique combination of spindle speed and feed rate was used, and a single file was generated. Drilling operations were performed sequentially, and tool wear was monitored by acquiring images every 10 holes. This ensured consistent wear tracking across all experimental conditions.

The complete experimental matrix is presented in Figure 3.6.

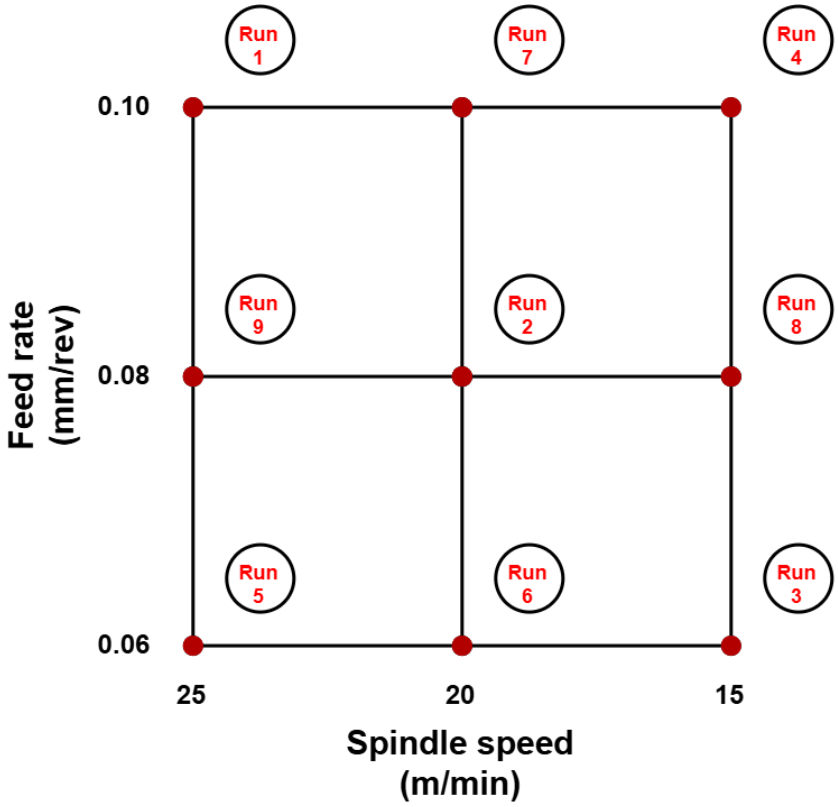


Figure 3.6: Final full-factorial experimental plan (3 feed levels × 3 speed levels).

This full-factorial structure ensures complete coverage of the parameter space, enabling the analysis of trends and the development of data-driven models for tool condition monitoring.

3.2.3 G-code programming and machining strategy

The machining program used in this work was generated using *Autodesk Fusion*. The workpiece geometry was first modelled in the CAD environment, after which the CAM module was used to define the drilling operations. For each experimental block, a toolpath was created to drill 300 holes with the same nominal diameter as the drill used in the corresponding DoE. After the initial post-processing, the generated G-code was manually edited to incorporate the specific requirements of the experimental procedure. The code used can be seen in Appendix A.

The post-processed and manually adjusted G-code included commands to activate and deactivate the digital input (DI) signal used for synchronisation. The DI signal was turned *on* at the beginning of each drilling cycle and turned *off* at the end, allowing precise temporal alignment between the sensor signals and the drilling events. Additionally, the program was configured to automatically pause after every 10 holes to allow the acquisition of tool wear images.

Two different drilling strategies were implemented, corresponding to the two DoEs:

- **Solid carbide drill DoE:** A conventional drilling cycle was used, producing holes with a depth of 12 mm, with internal coolant activated during cutting.
- **HSS drill DoE:** A deeper hole depth of 24 mm was chosen, and a peck drilling cycle was adopted to improve chip evacuation and reduce tool loading. The peck depth was set to 6 mm. In this configuration, flood coolant was used instead of internal coolant, matching the capabilities of the HSS drill.

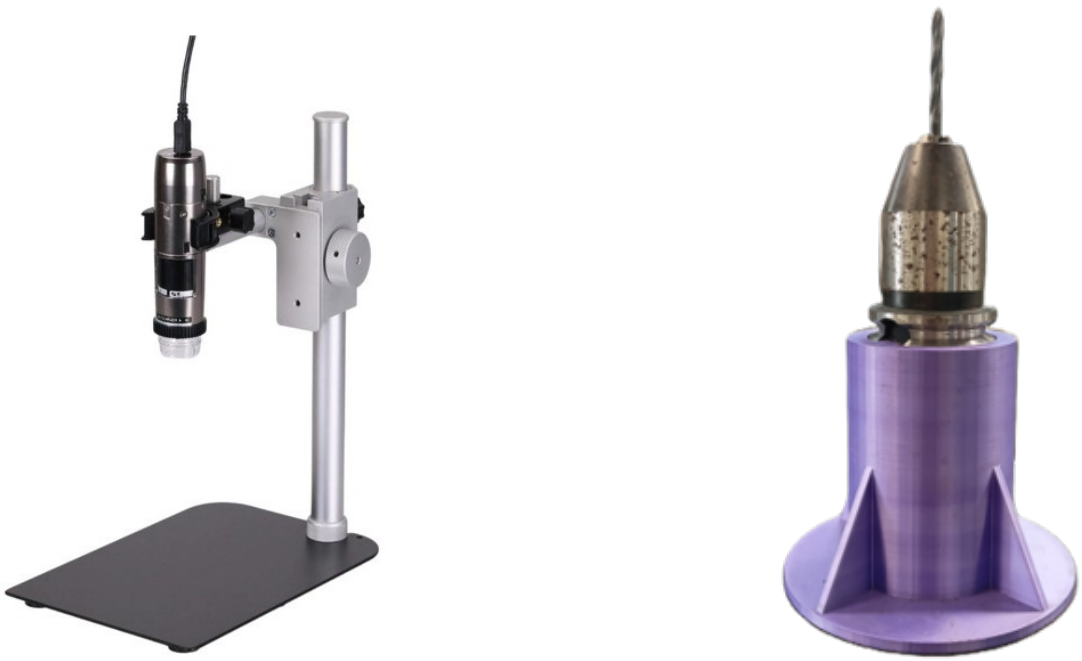
3.2.4 Tool wear measurement

The evaluation of tool wear was performed through periodic image acquisition and subsequent measurement of the flank wear land. The end-of-life criterion was based on ISO standard 8688-1, which defines a flank wear limit of $V_B = 0.3$ mm. Although this standard was developed for milling operations, no equivalent standard specifically defining tool-life criteria for drilling is available. Consequently, the criterion is commonly adopted in the literature for drilling applications, with the relevant definitions adapted to the drilling process.

Acquisition setup

A digital microscope *Dino-Lite Edge* [57] was used to capture the images. Prior to the measurements, a calibration procedure was carried out to determine the pixel-to-millimetre conversion factor. This was achieved by imaging a reference object with a known dimension, allowing the establishment of a precise scale for converting pixel distances into real physical units. The resulting scale for all measurements was 181.3382 pixels/mm.

To ensure repeatable positioning across acquisitions, a dedicated support fixture was designed and manufactured using a 3D printer. This fixture held the tool holder and drill in a fixed orientation directly under the camera, ensuring consistent alignment between measurement sessions. Figure 3.7 shows the digital microscope and the 3D-printed positioning fixture used throughout the experimental campaign.



(a) Dino-Lite Edge digital microscope.

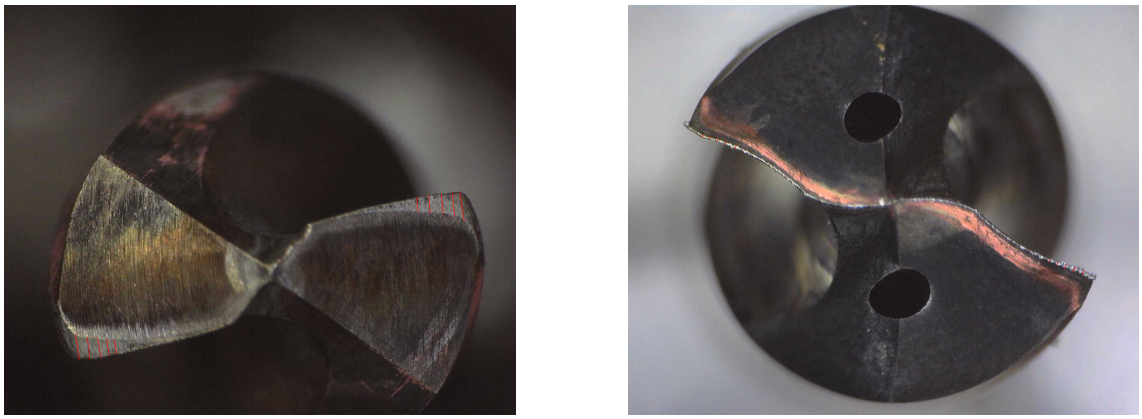
(b) 3D-printed fixture with drill in measurement position.

Figure 3.7: Image acquisition setup used for tool wear measurement.

Images of the drill were taken every 10 holes, always from a top view. The photographs were collected over several days, which resulted in noticeable variations in lighting conditions. These differences are visible in the images and represent a source of visual variability inherent to the acquisition process.

Measurement procedure

The wear measurements were performed using *ImageJ* software. The flank wear width V_B was measured according to the definition provided in ISO standard 8688-1 [21]. For each image, up to five measurements were taken on each flute whenever possible. In some cases, material adhered to the cutting edge prevented measurement on one side; in such situations, only the measurable side was considered. To reduce operator-induced variability, the individual measurements for each flute were averaged, and this average value was taken as the representative V_B for that acquisition point. Representative examples of acquired images with the corresponding wear measurements overlaid in red are shown in Figure 3.8.



(a) HSS drill flank wear measurement.

(b) Solid carbide drill flank wear measurement.

Figure 3.8: Example microscope images with V_B measurements performed in *ImageJ*.

Treatment of tool fracture events

In cases where the tool fractured before reaching the ISO wear limit (in 5 of the 9 runs this happened), the V_B value of the corresponding peck was set to 0.5 mm to ensure that the signals from that peck and all prior pecks are assigned to the worn tool class, enabling the model to detect this failure mode. Although assigning a specific V_B value to a fracture event is not physically meaningful, this approach is justified by the classification nature of the developed models, in which the relevant output is the wear class rather than the exact V_B magnitude.

The resulting wear values were used as ground truth labels for the analysis and model development presented in the following chapters.

3.3 Dataset generation and preparation

This section presents the datasets obtained during the experimental campaign. Two datasets were generated: one corresponding to the solid carbide drill tests and another corresponding to the HSS drill tests. The first dataset served mainly to understand the behaviour of the solid carbide tool, while the second dataset constitutes the main source of data used for tool wear modelling and TCM algorithm development.

3.3.1 Dataset: Solid carbide tools

The first dataset was acquired during the initial experimental plan using the solid carbide drill. Images of the tool were taken every 10 holes, and the flank wear V_B was measured following the procedure described in 3.2.4. The resulting wear curve 3.9 showed extremely slow wear progression: even after more than 1300 drilled holes, the measured V_B remained well below the ISO 8688-1 end-of-life limit of 0.3 mm.

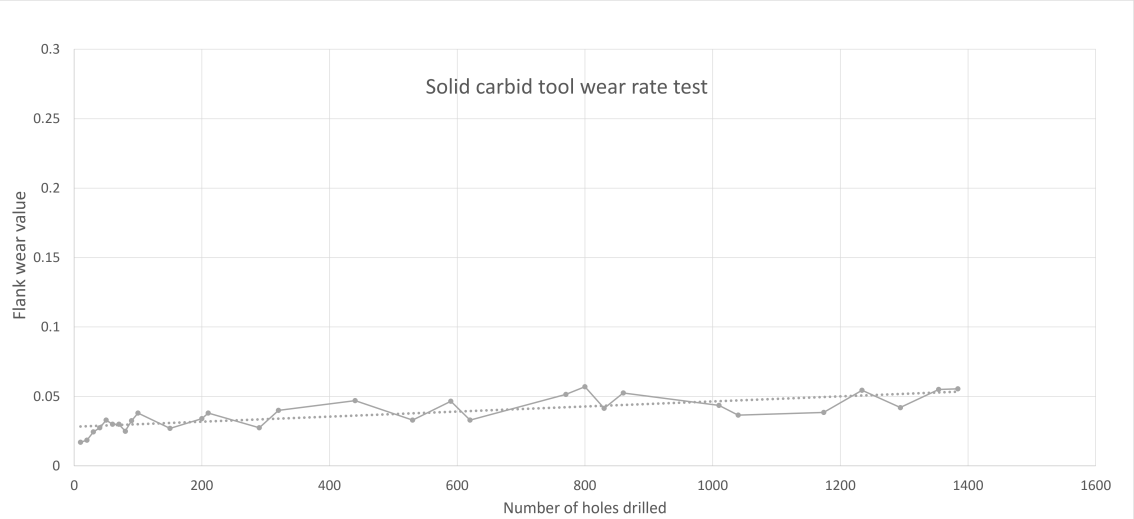


Figure 3.9: Wear measurements of a test run of a solid carbide drill to estimate the wear rate.

This behaviour is consistent with the high wear resistance of solid carbide tools when machining AISI 316L. However, the very low wear rate made it impractical to complete the full DoE within the available time and material. For this reason, the solid carbide dataset was discontinued, and the tool type was changed to HSS, enabling measurable wear progression within feasible experimental constraints.

3.3.2 Dataset: High speed steel tools

The second dataset corresponds to the full-factorial DoE performed with the HSS drill. This dataset contains all sensor signals (forces, currents, audio, accelerometer) and the corresponding tool wear measurements. Unlike the solid carbide case, the HSS drill exhibited measurable wear progression as shown in Figure 3.10, allowing the construction of a complete labelled dataset suitable for TCM model development.

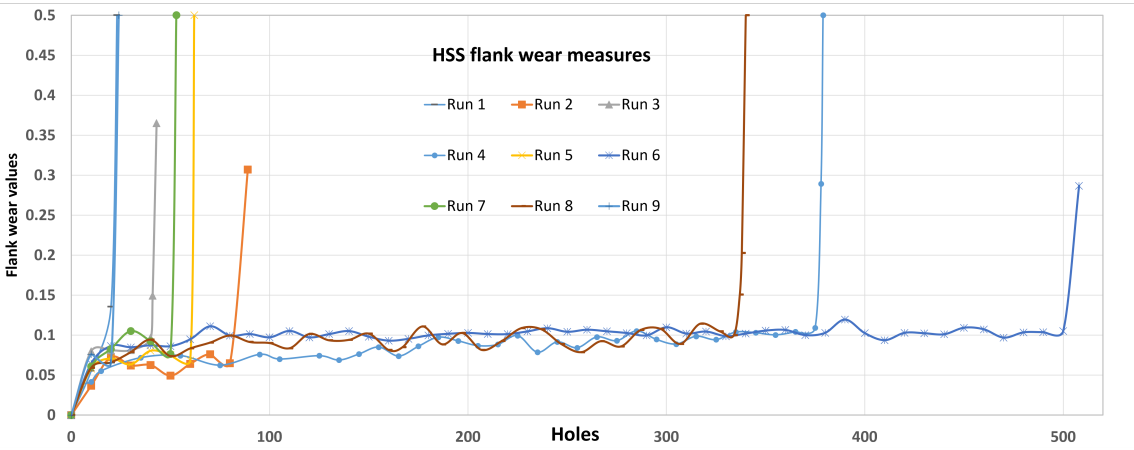


Figure 3.10: Wear measurements made in the HSS experiments.

3.3.3 Dataset labelling and class definition

Wear measurements were taken every ten holes, meaning that each block of ten consecutive holes shares a single measured V_B value. Since each hole contains up to four pecks, this results in up to forty pecks per measurement block with no direct individual label. To assign a wear value to each peck, linear interpolation was performed between consecutive V_B measurements, distributing wear monotonically across the intervening pecks.

Based on these continuous wear values, a three-class labelling scheme was defined. The class boundaries were established through a analysis of the ISO tool life criterion discussed before. The thresholds were defined through direct observation of the wear data. The three classes are defined as follows:

- **Class 0 - New:** $V_B < 0.07$ mm
- **Class 1 - Used:** $0.07\text{mm} \leq V_B < 0.12$ mm
- **Class 2 - Worn:** $V_B \geq 0.12$ mm

The thresholds of 0.07 mm and 0.12 mm correspond to the average locations of the inflexion points observed across the wear curves, marking the transitions between the initial running-in phase, the stable wear regime, and the accelerated wear phase. Figure 3.11 illustrates the holes at which the V_B measurements were taken, along with their distribution per Run and class. Figure 3.12 presents the class distribution of the pecks per file after interpolation.

To account for the irreversible nature of tool wear, a monotonicity constraint was enforced during labelling: once a tool crosses into a higher class, it cannot be relabelled back to a lower one, even if a later interpolated V_B value happens to fall below that class's threshold. A tool labelled Class 1, for example, stays Class 1 (or moves to Class 2) from that point onward, never back to Class 0. Flank wear is cumulative and does not reverse itself, so letting labels drop back down would just be tracking noise or interpolation artefacts rather than anything physically meaningful.

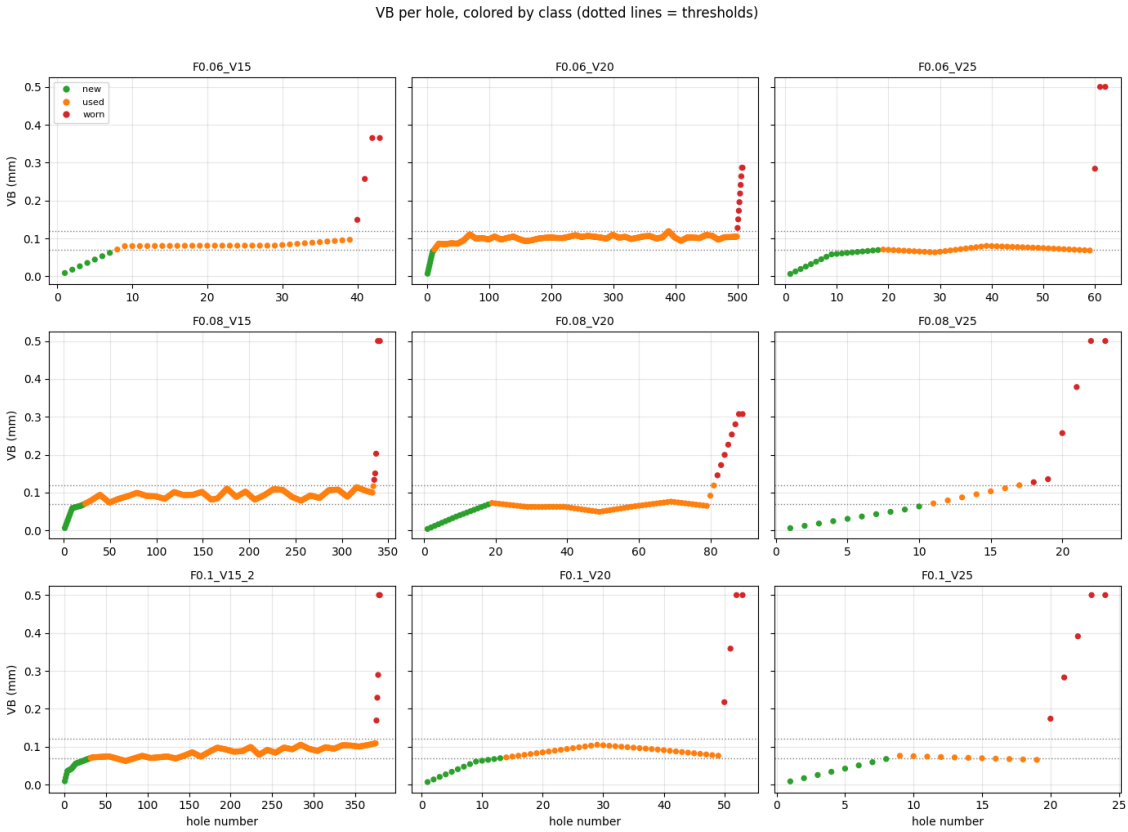


Figure 3.11: Measured holes per Run.



Figure 3.12: Distribution of the points per class per Run.

It is acknowledged that this boundary definition is not unique and that alternative threshold choices could yield different class distributions. Given that the dataset is derived from a limited number of experimental runs, this decision may have a significant influence on model performance and should be considered when interpreting the results presented in subsequent chapters.

3.3.4 Dataset structure

The dataset generated with the HSS drill is organised into a set of nine `.h5` files, each corresponding to one of the experimental conditions defined in the full-factorial DoE described in Figure 3.6. Each file therefore represents a complete experiment performed with a unique combination of spindle speed and feed rate as shown in Table 3.3.

Table 3.3: Run to file conversion.

Run	File
1	F0.10_V25
2	F0.08_V20
3	F0.06_V15
4	F0.10_V15 (F0.10_V15_2)
5	F0.06_V25
6	F0.06_V20
7	F0.10_V20
8	F0.08_V15
9	F0.08_V25

Inside each `.h5` file, the data are structured hierarchically to reflect the drilling process, as illustrated in Figure 3.13. The top level is divided into holes, and each hole is further subdivided into pecks. Each peck contains a complete set of synchronised sensor signals, including force components, current measurements, audio waveform, accelerometer channels, and the associated timestamps. This organisation allows direct access to the exact portion of the drilling process corresponding to each peck, facilitating segmentation, feature extraction, and label assignment.

The hierarchical structure is consistent across all nine experiments: every file follows the same organisation, with holes numbered sequentially and each hole containing up to four pecks.

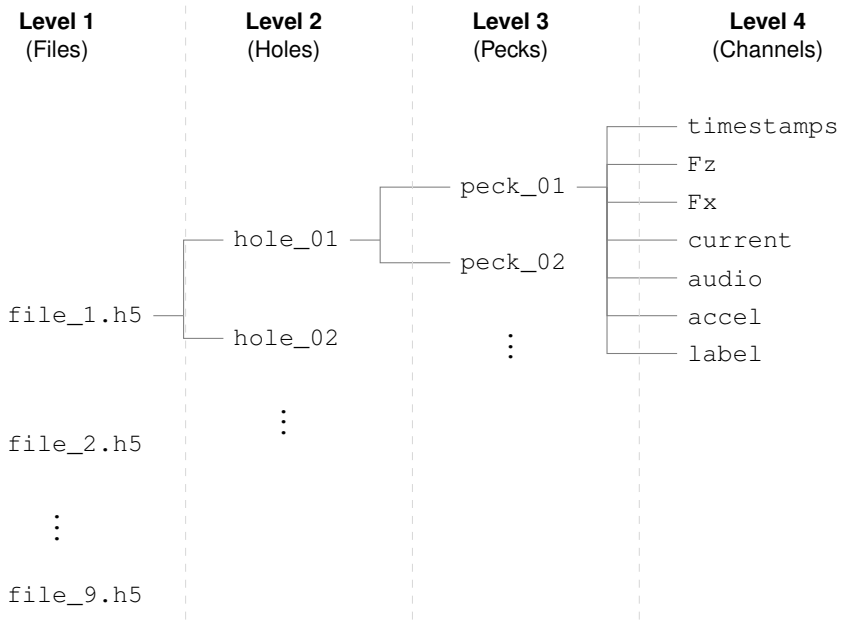


Figure 3.13: Hierarchical structure of one `.h5` experimental file. One path is fully expanded to show all four levels; all remaining nodes follow the same organisation.

Chapter 4

Methodological framework for tool condition classification

This chapter describes the complete methodological framework used to develop, train, and evaluate the proposed TCM models. Two parallel classification paths are implemented: a classical machine learning path based on handcrafted statistical and spectral features, and a deep learning path based on a multi-branch one-dimensional CNN operating directly on raw sensor signals. Both paths share the same data loading, class-imbalance handling, validation strategy, and evaluation metrics, ensuring a fair and direct comparison between them.

4.1 Creation of the datasets

The collected and stored data are processed through a preprocessing pipeline to transform the raw recordings into a structured dataset containing only signals corresponding to the steady-state cutting regime. Two dataset versions are generated from this pipeline. The first, referred to as the *raw dataset*, is obtained after the segmentation and selection of the relevant cutting periods, without applying any additional denoising. The second, referred to as the *denoised dataset*, undergoes additional signal processing steps aimed at removing unwanted noise and disturbances from the acquired signals.

4.1.1 Data cleaning and segmentation

Following the acquisition of sensor signals across all machining experiments, a series of processing steps were required to transform the raw recordings into a structured and usable dataset. The raw signals contain large portions of irrelevant data, including idle periods between holes, tool repositioning movements, and transient phenomena at the entry and exit of each cut. The segmentation pipeline described below was designed to isolate only the steady-state cutting regions from each peck.

1. **DI-based segmentation.** The first step removed all data recorded outside the active drilling process. A digital input (DI) signal was used as a synchronisation indicator, remaining active only during the execution of the drilling G-code. All time intervals in which the DI signal was inactive were discarded, retaining only the portions of the recording corresponding to actual machining activity.

2. **Detection of peck boundaries.** Within the retained intervals, each hole is composed of multiple pecks separated by tool retractions. To isolate each peck individually, the vertical force channel F_z was used as the detection signal. Since the dynamometer is mounted beneath the workpiece, F_z provides a direct measure of the axial cutting force, which rises sharply when the drill contacts the material and drops back to zero upon retraction. A moving average filter was first applied to F_z to suppress high-frequency noise while preserving the overall force envelope. A fixed threshold was then applied to the smoothed signal to detect the onset and termination of each peck. The detected boundaries, together with the smoothed F_z signal, the threshold level and the DI signal (yellow), are illustrated in Figure 4.2. The corresponding raw F_z signal, before the application of the moving average filter, is shown in Figure 4.1 for comparison.

3. **Extraction of steady-state cutting.** The boundaries identified in the previous step define the approximate start and end of each peck. However, the entry and exit regions of each peck contain transient phenomena that are not representative of the steady-state cutting process. To exclude these regions, a fixed percentage of the total peck duration was removed from both the beginning and the end of each detected interval. The remaining central portion, corresponding to the steady-state cutting regime, was retained as the final segmented signal for that peck.

Although the G-code defines four pecks per hole, the figures show five. This occurs because the tool's initial position is set 1 mm above the workpiece surface, leaving a short fifth peck after the four programmed ones to reach the final depth of 24 mm. This fifth peck is excluded from the dataset as it never reaches a steady-state cutting regime.

The output of this pipeline a collection of clean, well-segmented signal windows, one per peck, containing only the steady-state cutting behaviour. These windows form the basis for all subsequent feature extraction and model training steps described in the following chapters.

Figures 4.1 and 4.2 illustrate the application of the processing pipeline to the last three holes of a representative drilling run.

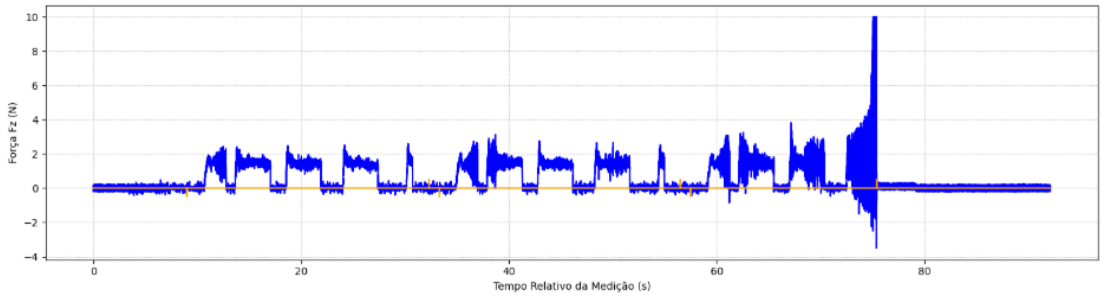


Figure 4.1: Raw F_z signal (in blue) recorded during a representative drilling sequence with 3 holes, before any filtering. Yellow line shows the digital input state that marks the beginning and end of a hole in the g-code.

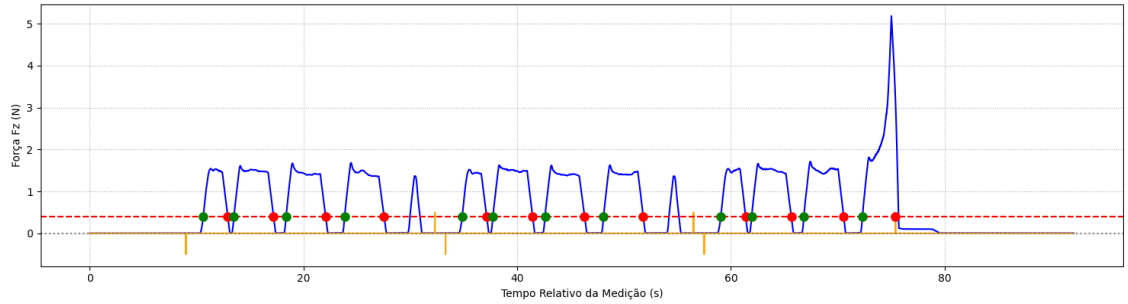


Figure 4.2: Peck boundary detection applied to the smoothed F_z signal. The moving average filter output (solid line in blue), the detection threshold (dashed line in red), and the identified peck start (green circles), end boundaries (red circles) and the digital input state (yellow line) are shown for the representative drilling sequence with 3 holes.

After applying this pipeline to the complete set of collected data, the first dataset, referred to as the *raw dataset*, is obtained.

4.1.2 Denoising pipeline

Raw sensor signals acquired during machining contain substantial noise contributions unrelated to tool wear: ambient electrical interference present even when the machine is idle, the spindle's own mechanical and electrical signature at its operating speed and harmonics, and intermittent impulsive transients caused by electrical spikes or mechanical events unrelated to the cutting process. If left uncorrected, these contributions can dominate the extracted features and cause a model to learn cutting-condition artefacts rather than wear-related signal changes.

To mitigate this, a sequential processing pipeline was developed based on techniques identified during the literature review, and applied to each peck signal before storage in the denoised dataset, leaving the raw dataset unchanged. The pipeline begins with two channel-level corrections: a fourth-order zero-phase Butterworth high-pass filter at 10 Hz removes the DC offset introduced by gravity from all three accelerometer channels, and the force signals are scaled by their respective amplifier gains ($F_x \times 100$, $F_y \times 100$, $F_z \times 500$) to convert raw DAQ readings into physical units.

Four denoising stages follow these corrections applied per channel; Table 4.1 summarises which stages are applied to each channel and the rationale for any exception.

S1 - Spectral Subtraction of Base Noise

The first stage removes the stationary background noise floor present in all signals regardless of spindle state. A reference signal is recorded with the machine fully idle: spindle off, no movement, and no coolant. The spectral power of this reference is estimated and subtracted from each cutting signal using the spectral subtraction method. For a signal short-time Fourier transform $Z_{\text{sig}}(f,t)$ and a noise power estimate $\bar{N}^2(f)$ computed as the median power of the reference spectrum across time frames, the cleaned spectral power is obtained as:

$$|Z_{\text{clean}}(f,t)|^2 = \max\left(\left|Z_{\text{sig}}(f,t)\right|^2 - \alpha\bar{N}^2(f),\ \beta\left|Z_{\text{sig}}(f,t)\right|^2\right) \tag{4.1}$$

Where $\alpha = 1.0$ is the over-subtraction factor and $\beta = 10^{-4}$ (-40 dB) is the spectral floor that prevents the cleaned signal from going to zero in frequency bins where the noise estimate exceeds the signal. The phase of the original signal is preserved, and the cleaned signal is reconstructed via the inverse STFT. This operation is applied to the accelerometer channels a_x, a_z, the audio channel, and the current channels C_{RMS} and C_1.

S2 - Air-Cut Matched Subtraction

The second stage removes the spindle's own mechanical and electrical signature from each cutting signal. A dedicated air-cut recording was performed for each combination of spindle speed and feed rate used in the experiments: the drill executes the same G-code trajectory but above the workpiece surface, so no material is being removed. The resulting signal therefore contains only the spindle's contribution, bearing noise, motor harmonics, and tool vibration in air at that exact operating condition.

The same spectral subtraction of Equation 4.1 is applied a second time, now using the air-cut recording matched to each file's spindle speed and feed rate as the noise reference. This step is applied to a_x, a_z, the audio channel, and C_{RMS}. It is deliberately omitted from C_1, where empirical evaluation showed that air-cut subtraction reduced inter-class separability, and from the force channels, for which the air-cut reference is physically irrelevant since the dynamometer is mounted beneath the workpiece and records no meaningful force during an air cut.

S3 - FFT Peak-Targeted Subtraction

Even after stages S1 and S2, narrowband tonal components at spindle harmonic frequencies can remain in the spectrum. These are identified and selectively attenuated in the frequency domain. For each channel, the FFT amplitude spectrum of the matched air-cut reference is computed, and peaks are detected as bins exceeding the 99th percentile of the amplitude distribution with a

minimum prominence of 30% of the threshold. Each detected peak frequency f_k with amplitude A_k is then attenuated in the cutting signal's spectrum over a ± 2 bin neighbourhood:

$$A_{\text{clean}}(f) = \begin{cases} \max(A(f) - A_k,\ 0.1 \cdot A(f)) & \text{if } |f - f_k| \leq 2\Delta f \\ A(f) & \text{otherwise} \end{cases} \tag{4.2}$$

Where Δf is the FFT frequency resolution. The original phase is preserved, and the signal is reconstructed via the inverse FFT. This stage is applied to a_x, a_z, audio, C_1, and C_{RMS}.

S4 - Hampel Spike Removal

The final stage operates in the time domain. It targets impulsive outliers: isolated samples that deviate sharply from the local signal behaviour due to electrical transients or mechanical impacts unrelated to wear. A Hampel identifier is applied using a sliding window of length $w = 7$ samples (extended to $w = 11$ for the audio channel given its higher sampling rate). Within each window, the local median m_i and the median absolute deviation (MAD) are computed, and any sample caught by the filter is replaced by the local median m_i. Stage S4 is the only stage applied to all channels, including the force signals and the a_y accelerometer axis.

Table 4.1: Denoising stages applied per sensor channel.

Channel	S1	S2	S3	S4	Note
a_x, a_z	✓	✓	✓	✓	Full pipeline
a_y	-	-	-	✓	S1–S3 introduce artefacts
Audio	✓	✓	✓	✓	$w = 11$ for Hampel
C_{RMS}	✓	✓	✓	✓	Full pipeline
C_1	✓	-	✓	✓	S2 reduced class separability
F_x, F_y, F_z	-	-	-	✓	Air-cut reference not applicable

After applying the denoising pipeline to the first dataset, the second dataset, referred to as the *denoised dataset*, is obtained.

4.2 Feature extraction

The two classification paths require fundamentally different representations of the peck signal.

4.2.1 Classical path: Handcrafted features

For the classical machine learning models, each peck is summarised into a fixed-length numerical vector. Twelve features are computed per active sensor channel, spanning three domains: time, frequency, and a combined temporal-spectral domain. This gives a total feature vector of length $12 \times N_{\text{channels}}$, summarised in Table 4.2.

Table 4.2: Handcrafted features computed per sensor channel.

Domain	Feature	Description
Time domain	RMS	Overall signal energy level
	Standard deviation	Spread of the signal around its mean
	Kurtosis	Fourth standardised moment
	Peak-to-peak amplitude	Range of the signal
Frequency domain	Spectral centroid	Frequency-weighted average of the PSD
	Low-band energy	Integrated PSD in $[0, 0.1 \cdot f_{\text{Nyquist}})$
	Mid-band energy	Integrated PSD in $[0.1, 0.5) \cdot f_{\text{Nyquist}}$
	High-band energy	Integrated PSD in $[0.5, 1.0] \cdot f_{\text{Nyquist}}$
Temporal-spectral domain	Spectral entropy	Entropy of the normalised PSD
	Wavelet detail-energy ratio	Finest-detail-band energy over total energy
	Spectral kurtosis	Excess kurtosis of each frequency bin's magnitude over time, from the spectrogram, averaged across bins
	Wavelet packet energy entropy	Entropy of the energy split across the level-3 wavelet-packet nodes

4.2.2 CNN path: Raw windowed signal

For the CNN, no hand-engineered features are computed. Instead, each peck is divided into overlapping time windows of fixed duration $T_{\text{win}} = 2.0\,\text{s}$ with 25 % overlap (stride $= 0.25 \cdot T_{\text{win}}$). This window length was chosen because it covers most pecks in their entirety: only a small minority of pecks are shorter than T_{win} and are discarded outright, so the window converges toward using the full peck signal rather than a short fragment of it, bringing the CNN's input closer to the amount of signal the classical path already uses, since its handcrafted features are computed over the whole peck. Each window is passed directly to the network as a raw signal. Because sensor channels operate at different sampling rates, the same T_{win} produces different numbers of samples per channel: 10 000 samples for DAQ channels, 53 332 for the accelerometer, and 96 000 for audio. Windows shorter than one full T_{win} are discarded.

Since the network operates on windows but the target label is defined at peck level, the predicted class probabilities of all windows belonging to the same peck are averaged and the final class is determined by the argmax of the mean probability vector. In the common case where a peck produces a single window, this reduces to using that window's prediction directly; on the minority of pecks long enough to produce two or more windows, the aggregation combines their predicted probabilities before taking the argmax, rather than picking one window over the other. This aggregation ensures that CNN predictions are directly comparable to the classical baseline, which produces one prediction per peck.

4.3 Class imbalance handling

The dataset is structurally imbalanced: the transition from new to used tool (class 0 to class 1) happens early in the tool life, whereas the worn class (class 2) represents only the final portion of each experiment, making it the rarest class by a significant margin. The natural class distribution across all files is shown in Figure 3.12

Two mutually exclusive strategies are evaluated as a grid axis to address this imbalance. They are never combined. In both cases the strategy applies only to the training data; the test file is always used in full and is never modified.

4.3.1 Undersampling

The first strategy reduces the dominance of the majority class by discarding training rows. A fixed budget N is distributed across the three classes as evenly as possible using a water-filling algorithm: rows are allocated one at a time to whichever class currently has the fewest allocated rows, until the budget N is exhausted, or all classes are fully represented.

An alternative mode is also evaluated in which all rows from the two minority classes are kept intact, and only the majority class is subsampled to a target count equal to a small multiple of the average minority class size. This prevents discarding any rare worn samples while still reducing the majority class dominance. For this study, the value N was chosen as 50. The effect of this strategy on the training set class distribution is illustrated in Figure 4.3

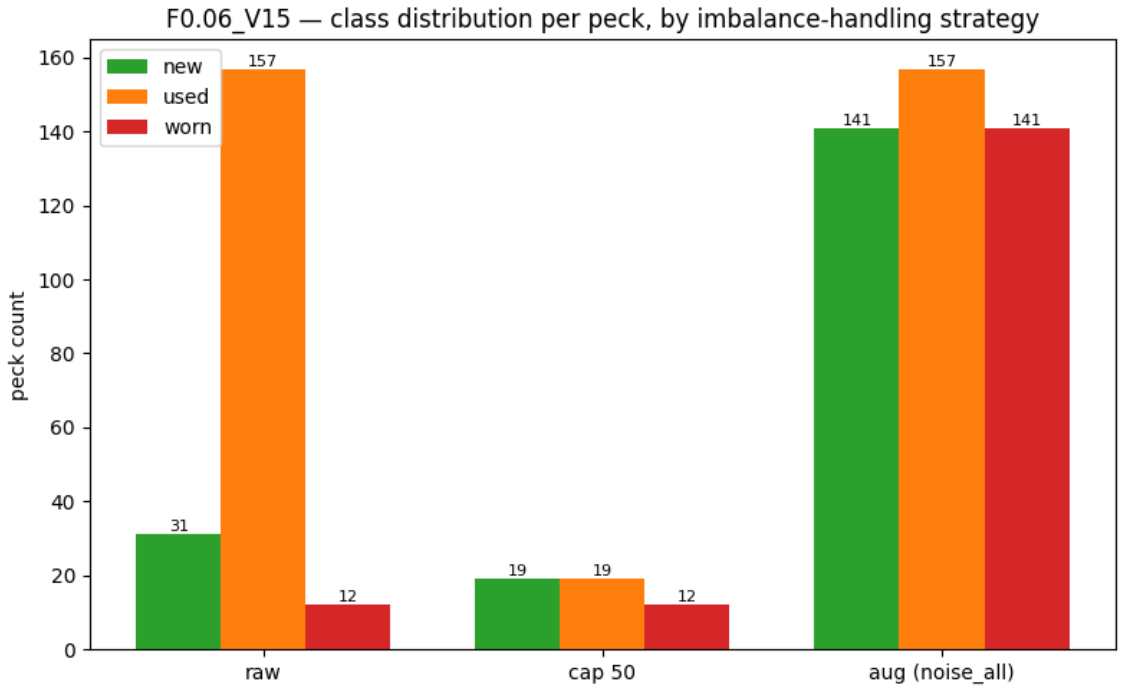


Figure 4.3: Training set class distribution before and after each imbalance handling strategy, shown for a representative experiment.

4.3.2 Augmentation

The second strategy grows the minority classes by creating synthetic training samples for all channels rather than discarding majority ones. Crucially, perturbation is applied to the raw signal waveform, and features are recomputed from the perturbed signal. This is physically meaningful: adding noise directly to a precomputed RMS value would produce a number that no real drill signal could generate. Perturbing the waveform and recomputing keeps every feature physically consistent with every other.

Four perturbation methods are implemented:

- **Gaussian noise** additive noise scaled to $\sigma_{\text{noise}} = 0.02 \cdot \sigma_{\text{signal}}$, simulating sensor noise variation.
- **Amplitude scaling** multiplication by a uniform random factor in $[0.90,\ 1.10]$, simulating slight changes in material hardness or depth of cut.
- **Time shift** circular roll by up to $\pm 5\,\%$ of the signal length, reflecting that drilling pecks are not phase-synchronised across holes.
- **Channel dropout** a contiguous block of up to $10\,\%$ of the signal is zeroed, simulating a momentarily blocked or disconnected sensor.

Minority classes are synthesised until each reaches $90\,\%$ of the majority class count. The normalisation parameters (Section 4.4) are fitted on the real training rows before augmentation; synthetic rows are transformed by those parameters but do not influence them.

The effect of this strategy is illustrated in Figure 4.3.

4.4 Normalisation

All sensor signals and feature vectors are standardised using z-score normalisation, transforming each channel or feature x according to:

$$\hat{x} = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}} \tag{4.3}$$

where μ_{train} and σ_{train} are the mean and standard deviation estimated from the training data of each cross-validation fold. Global normalisation is deliberately used rather than per-run normalisation. The amplitude of sensor signals carries information about the cutting condition, namely the combination of feed rate and spindle speed, and these differences are intentional experimental signals rather than artefacts. Normalising within each run would erase this information, reducing the model's ability to distinguish between cutting conditions. The normalisation parameters are never computed from the test file, ensuring no information leakage across folds.

4.5 Model architectures

4.5.1 Random forest

The classical baseline uses a Random Forest classifier. An ensemble of 300 decision trees is trained on the handcrafted feature vectors described in Section 4.2.1. To address class imbalance at the model level independently of the data-level strategies described in Section 4.3, inverse class-frequency weighting is applied: each tree assigns a higher misclassification penalty to the minority classes, so that errors on worn samples are penalised proportionally more than errors on the dominant used class. The model produces one prediction per peck directly from its feature vector.

4.5.2 Extreme gradient boosting

As a second classical model, gradient boosting is used through the Extreme gradient boosting (XGBoost) implementation. Rather than averaging independent trees as in the Random Forest, XGBoost builds trees sequentially, with each new tree trained to correct the residual errors left by the ensemble so far. The model is trained for 300 boosting rounds with a maximum tree depth of 5 and a learning rate of 0.1. At each round, only 90% of the training samples and 90% of the features are sampled to build the next tree, which keeps successive trees from becoming too correlated and helps limit overfitting. Trees are grown using the histogram-based method for faster training, and the model is optimised with a softmax-based multi-class objective, producing a probability distribution over the three wear classes for each peck. Unlike the RF, no class weighting is applied at the model level for XGBoost; class imbalance is therefore handled exclusively through the data-level strategies described in Section 4.3.

4.5.3 Multi-Branch 1D-CNN

Sensor grouping and branch structure

The CNN receives raw windowed signals rather than handcrafted features. Because the sensor channels originate from physically distinct instruments operating at different sampling rates, a single-branch network is not straightforward: a 1-second window produces 5 000 samples from the DAQ channels, 26 666 from the accelerometer, and 48 000 from the microphone. These cannot be concatenated into a single input tensor without resampling, which would either discard high-frequency content or introduce artefacts.

Instead, channels are grouped by sensor type, and an independent branch processes each group. Four groups are defined: force (F_x, F_y, F_z), current (C_{RMS}, C_1), accelerometer (a_x, a_y, a_z), and audio. Force and current channels share the same DAQ sampling rate of 5 kHz but are still assigned to separate branches, since they carry physically different information: cutting force in Newtons versus motor current in Amperes that should not share convolutional filters.

This grouping also enables a channel-set grid axis, so we can train the model on all possible sensor combinations.

Branch architecture

Each branch is a small 1D-CNN tower that takes a tensor of shape $[B, C_g, L_g]$ batch size, number of channels in the group, and number of samples in the window for that group's sampling rate and produces a fixed-length embedding vector of dimension 64 regardless of the input length L_g. The architecture of each branch is identical and is summarised in Table 4.3.

Table 4.3: Architecture of each CNN branch. C_g is the number of input channels for the group; width doubles after the first block.

Layer	**Operation**	**Filters / Size**	**Output**
Conv block 1	Conv1d + BatchNorm + ReLU	32 filters, kernel 7	$[B, 32, L_g]$
	MaxPool1d	stride 4	$[B, 32, L_g/4]$
Conv block 2	Conv1d + BatchNorm + ReLU	64 filters, kernel 5	$[B, 64, L_g/4]$
	MaxPool1d	stride 4	$[B, 64, L_g/16]$
Conv block 3	Conv1d + BatchNorm + ReLU	64 filters, kernel 3	$[B, 64, L_g/16]$
Pooling	AdaptiveAvgPool1d	output length 8	$[B, 64, 8]$
Head	Flatten + Linear + ReLU	$64 \times 8 \to 64$	$[B, 64]$

The adaptive average pooling layer in the final step collapses the temporal dimension to a fixed length of 8 regardless of the input length L_g. This is what allows branches with different sampling rates to each produce a 64-dimensional vector of equal size that can then be concatenated.

Fusion and classification head

The output vectors from all active branches are concatenated into a single vector of dimension $64 \times N_{\text{branches}}$, where N_{branches} is the number of active sensor groups. This fused vector is passed through a two-layer MLP:

$$\mathbf{z} = \text{ReLU}(W_1 [\mathbf{v}_1 \| \cdots \| \mathbf{v}_N] + b_1), \quad \hat{\mathbf{y}} = W_2 \, \text{Dropout}_{0.3}(\mathbf{z}) + b_2 \tag{4.4}$$

where $W_1 \in \mathbb{R}^{128 \times 64N}$, $W_2 \in \mathbb{R}^{3 \times 128}$, and $\|$ denotes concatenation. The output $\hat{\mathbf{y}}$ contains the raw logits for the three wear classes.

Loss functions

The loss function used is Weighted cross-entropy, which assigns a higher penalty to misclassified samples from rarer classes. The per-class weight is set to the inverse class frequency, normalised so that the average weight remains equal to one.

Training procedure

The network is trained using the Adam optimiser with a learning rate of 10^{-3} and a batch size of 32. The number of training epochs is fixed per configuration and determined in advance using an epoch-finder procedure: a single probe run trains on seven files while monitoring validation loss on the eighth, and the epoch at which validation loss is minimised is used as the fixed budget for that configuration's full grid run.

No validation set is used during the grid training itself. This is a deliberate design choice that allows the CNN to train on all eight training files per fold, exactly as the classical baseline does, ensuring that the two models are trained on identical data and the comparison between them is honest.

4.6 Validation strategy

4.6.1 Run-level cross validation - Split A

Split A implements a Leave-One-Out cross-validation (LOOCV) at the run level. In each of the nine folds, one run (corresponding to one combination of parameters) is held out as the test set; the remaining eight files form the training set. The training set may be capped or augmented; the test file is always used in full and is never modified.

This split tests cross-condition generalisation: the model trained on eight cutting conditions must classify tool wear on a ninth condition it has never seen. This reflects the realistic industrial scenario in which a TCM system must operate across a range of machining parameters.

4.6.2 Hole-level cross validation - Split B

Split B reserves 15 % of each class's holes per run as a held-out test set, drawing from every run simultaneously. Holes are stratified by class before selection, so every class and every run contributes test holes; a hole is never present in both training and test. The test set retains its natural class distribution no rebalancing is applied so the dominant class remains dominant and accuracy diverges from balanced accuracy, making macro-F1 the only reliable headline number.

Because training and test data are drawn from the same set of cutting conditions, this split does not test cross-condition generalisation and its scores are expected to be substantially higher than Split A. It is included as an optimistic in-distribution upper bound and as a diagnostic: a large gap between Split B and Split A performance indicates that the model is sensitive to unseen cutting conditions rather than learning wear-related signal changes that transfer across conditions.

An important caveat applies specifically to this split: holes from the same run share the same tool, wear trajectory, and denoising references, so training and test holes within a run are not fully independent; part of Split B's score should therefore be read as reflecting this within-run correlation rather than condition-level generalisation alone.

4.7 Experimental grid

Rather than evaluating a single fixed configuration, all experiments are Structured as a systematic sweep over the axes listed in Table 4.4. Each unique combination of axis values constitutes a separate experiment, and the full sweep allows the contribution of each axis to classification performance to be measured independently.

Table 4.4: Experimental grid axes and their evaluated values.

Axis	**Values**
Model	Random Forest; XGBoost; Multi-branch 1D-CNN
Peck selection	all pecks
Preprocessing stage	Raw; denoised
Channel set	all
Imbalance strategy	None; Undersampled; Augmented
Validation split	Split A (run-level); Split B (hole-level)

For RF and the CNN, "None" refers only to the data-level strategy; both models still apply inverse-frequency loss weighting internally; for XGBoost, "None" means no correction of any kind

4.8 Evaluation metrics

For each class $c \in \{0(New), 1(Used), 2(Worn)\}$, let TP_c, FP_c, and FN_c denote the true positives, false positives, and false negatives for that class. Precision and recall are defined per class as

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c}, \qquad \text{Recall}_c = \frac{TP_c}{TP_c + FN_c}$$

Precision_c is the share of predictions of class c that were correct; recall_c is the share of true class-c instances that were correctly identified. Overall accuracy is the share of all predictions that were correct, regardless of class:

$$\text{Accuracy} = \frac{\sum_c TP_c}{N}$$

where N is the total number of test instances.

The primary metric throughout all experiments is macro-averaged F1 (macro-F1), built from the per-class precision and recall above:

$$\text{Macro-F1} = \frac{1}{3} \sum_{c \in \{0,1,2\}} F_{1,c} = \frac{1}{3} \sum_c \frac{2 \cdot \text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}$$

Macro-averaging treats all three classes with equal weight, meaning that correct detection of the rare worn class (class 2) contributes as much to the score as correct detection of the dominant used class (class 1). This is appropriate for the TCM context, where a missed worn tool is operationally more costly than a misclassified new tool.

4.9 Hardware and software

All experiments described in this work were carried out on a single workstation, whose hardware and software configuration is summarised in Table 4.5.

Table 4.5: Hardware and software configuration.

Component	Specification
CPU	Intel Core i7-12700 (20 threads)
RAM	62 GB
GPU	NVIDIA GeForce RTX 3070 (8 GB VRAM)
GPU driver	580.173.02
Python	3.10.12
PyTorch	2.10.0 (CUDA 12.8)
scikit-learn	1.7.2
XGBoost	3.2.0

Chapter 5

Results and discussion

This chapter presents and discusses the results obtained from evaluating the three classification models developed in this work: the Random Forest and XGBoost baselines, and the multi-branch 1D-CNN. All three were trained and tested on the same experimental grid, spanning both dataset stages, both training splits, and all three class-imbalance strategies.

5.1 Overall results

The results obtained in Table 5.1 across the full experimental grid: every combination of model, dataset stage, training split, and class-imbalance strategy evaluated in this work, together with the window and overlap settings used for the CNN. Results are ordered by macro F1 rather than accuracy, due to class imbalance, accuracy alone can mask poor performance on the minority *worn* class. This table serves as a starting point for the more detailed analysis that follows: the best-performing configurations are examined individually in Section 5.2, while the effect of each grid dimension on performance is discussed afterwards.

Table 5.1: Experimental grid and classification performance, ordered by macro F1.

Model	Stage	Imb. handling	Split	Window/overlap	Macro F1	Accuracy
CNN	Raw	None	B	2.0 s / 25%	**90.19%**	96.52%
XGBoost	Denoised	Augm	B		**88.13%**	95.95%
CNN	Denoised	None	B	2.0 s / 25%	86.57%	94.73%
XGBoost	Denoised	None	B		86.49%	95.53%
XGBoost	Raw	Augm	B		84.13%	94.60%
XGBoost	Raw	None	B		84.01%	94.70%
RF	Denoised	Augm	B		**81.40%**	91.48%
RF	Raw	None	B		79.89%	91.69%
RF	Denoised	None	B		79.84%	91.59%
CNN	Raw	Augm	B	2.0 s / 25%	79.68%	94.73%
RF	Raw	Augm	B		78.62%	92.52%
CNN	Denoised	Augm	B	2.0 s / 25%	77.16%	91.36%
XGBoost	Raw	Augm	A		**63.26%**	88.28%
XGBoost	Raw	None	A		63.18%	88.58%
XGBoost	Denoised	None	A		62.50%	88.99%
RF	Raw	None	A		**61.69%**	86.74%
XGBoost	Denoised	Augm	A		61.48%	87.12%
RF	Raw	Augm	A		61.40%	88.33%
CNN	Raw	Cap 50	A	2.0 s / 25%	**59.48%**	86.11%
RF	Denoised	None	A		57.39%	84.80%
CNN	Raw	None	A	2.0 s / 25%	57.26%	84.58%
RF	Denoised	Augm	A		55.22%	81.18%
CNN	Raw	Augm	A	2.0 s / 25%	55.00%	87.46%
CNN	Denoised	Augm	A	2.0 s / 25%	54.71%	85.50%
CNN	Denoised	Cap 50	B	2.0 s / 25%	52.97%	71.65%
CNN	Denoised	None	A	2.0 s / 25%	52.34%	83.40%
RF	Raw	Cap 50	B		50.95%	67.39%
XGBoost	Denoised	Cap 50	B		49.94%	65.42%
CNN	Denoised	Cap 50	A	2.0 s / 25%	49.71%	76.36%
RF	Denoised	Cap 50	B		49.59%	65.84%
CNN	Raw	Cap 50	B	2.0 s / 25%	48.62%	66.07%
RF	Raw	Cap 50	A		48.05%	69.52%
XGBoost	Raw	Cap 50	B		47.77%	64.07%
RF	Denoised	Cap 50	A		46.04%	67.39%
XGBoost	Raw	Cap 50	A		44.10%	63.70%
XGBoost	Denoised	Cap 50	A		41.71%	59.74%

5.2 Best-performing configurations

For each model, this section reports the single configuration that achieved the highest macro-F1, together with its confusion matrix and per-class metrics. For the CNN, the corresponding training curve is also included. Results are presented separately for Split A and Split B, followed by a comparison of what changes between the two.

Split A

Split A evaluates each model on cutting parameter combinations absent from the training set, testing generalisation rather than interpolation within known conditions, as described in Section 4.6.1. The best results for each model are presented in Table 5.2. The corresponding confusion matrices and per-class performance metrics for XGBoost, RF, and the CNN are shown in Figures 5.1, 5.2, and 5.3, respectively, while the CNN training curve is presented in Figure 5.4.

Table 5.2: Best-performing configuration per model on Split A.

Model	Stage	Imb. handling	Macro F1	Accuracy
XGBoost	Raw	Augm	63.26%	88.28%
RF	Raw	None	61.69%	86.74%
CNN	Raw	Cap 50	59.48%	86.11%

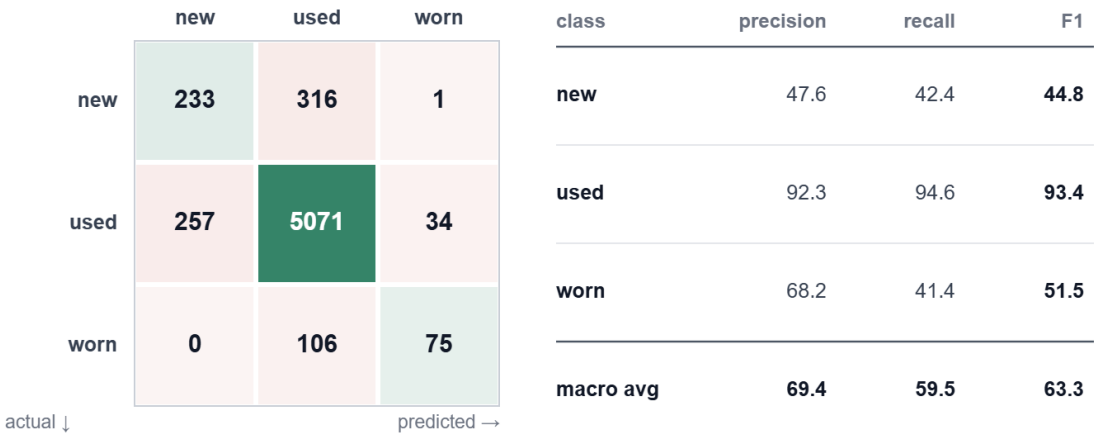
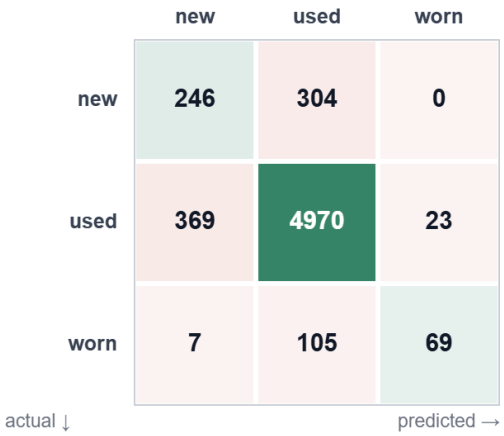
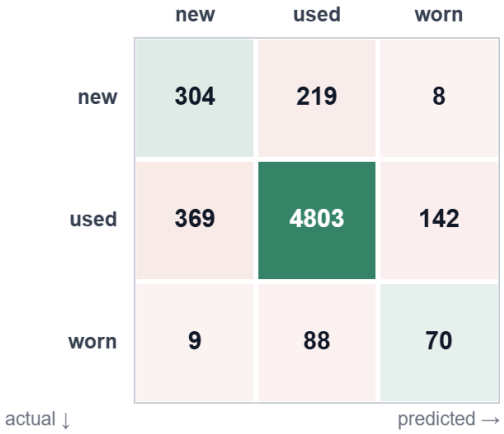


Figure 5.1: Confusion matrix and per-class metrics for XGBoost's best-performing configuration on Split A.



class	precision	recall	F1
new	39.5	44.7	**42.0**
used	92.4	92.7	**92.5**
worn	75.0	38.1	**50.5**
macro avg	**69.0**	**58.5**	**61.7**

Figure 5.2: Confusion matrix and per-class metrics for Random Forest's best-performing configuration on Split A.



class	precision	recall	F1
new	44.6	57.3	**50.1**
used	94.0	90.4	**92.2**
worn	31.8	41.9	**36.2**
macro avg	**56.8**	**63.2**	**59.5**

Figure 5.3: Confusion matrix and per-class metrics for the CNN's best-performing configuration on Split A.

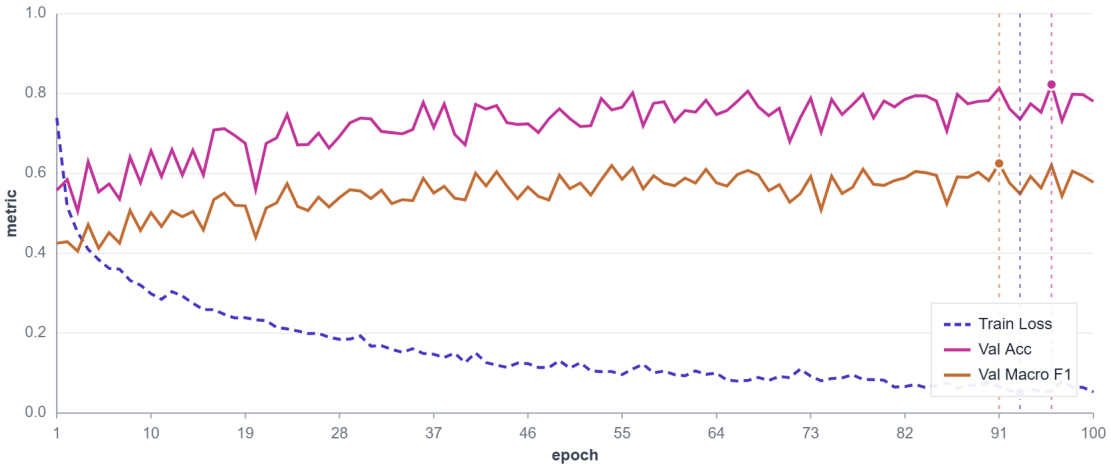


Figure 5.4: Training curve for the CNN's best-performing configuration on Split A.

On unseen parameters, XGBoost generalises best, with RF close behind and the CNN third. The CNN's best Split A configuration reaches 59.48% macro F1, only 3.78 % behind XGBoost and 2.21 % behind RF. The per-class breakdown in Figure 5.3 shows where the CNN still falls short: its worn-class F1 of 36.2% remains the weakest of the three models on the rarest and most critical class, against 51.5% for XGBoost and 50.5% for RF, although its new and used class scores are broadly comparable to the other two. The CNN's generalisation gap to the classical models is concentrated almost entirely in detecting the worn tool state, not in the more common classes. The training curve in Figure 5.4 is consistent with this reading: training loss decreases smoothly across the full 100 epochs, while validation accuracy and macro F1 peak earlier and then drift back down by the final epoch, consistent with overfitting.

After analysing these results, Figure 5.5 was created to identify where the top configurations fail.

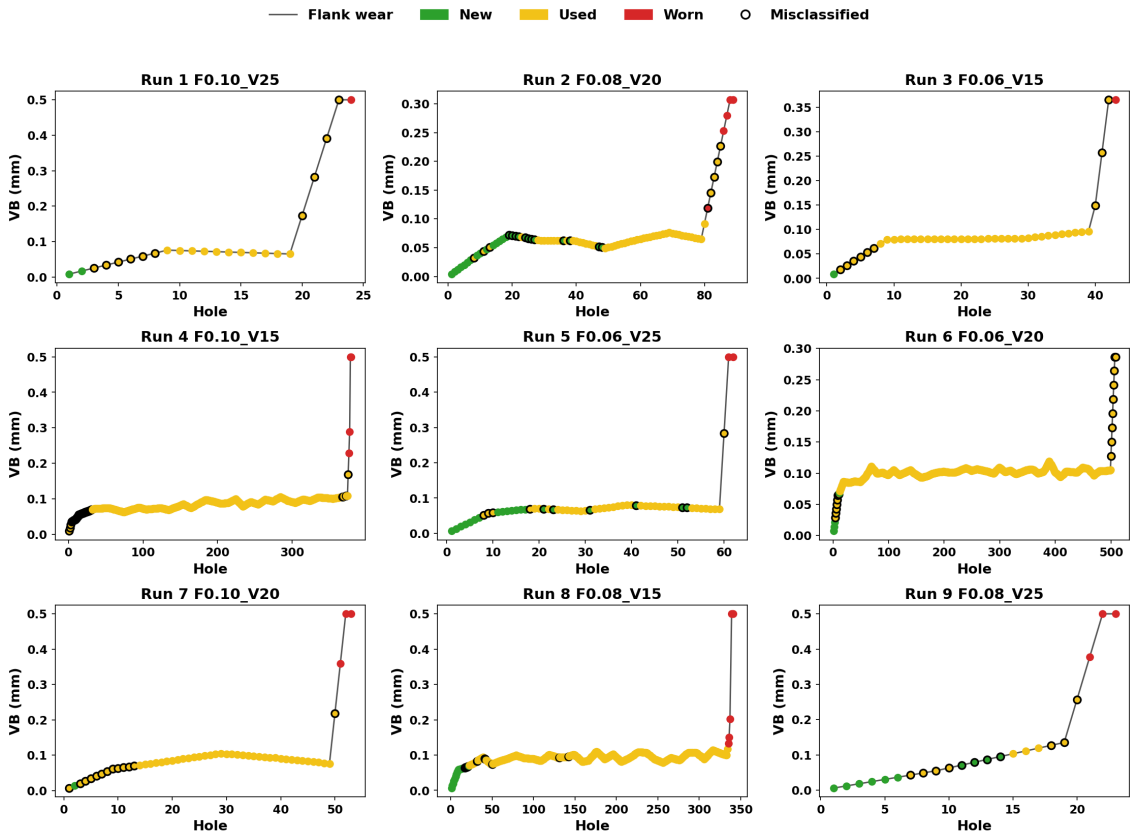


Figure 5.5: Real V_b vs predicted class in the best configuration of split A.

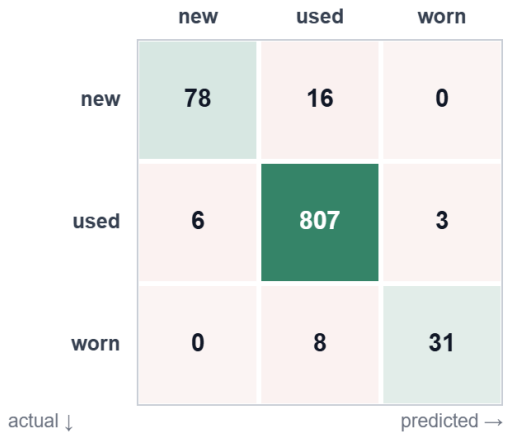
Figure 5.5 overlays XGBoost's predicted class on the actual flank-wear curve for each held-out run, with misclassified holes ringed in black. The errors are not confined to the borders of a class threshold. As can be seen more explicitly in Run 5 and Run 2, the misclassified holes extend well into the class 1 (used) plateau rather than sitting only at its boundaries, meaning the model is confusing entire stretches of a class, not just the ambiguous transition holes.

Split B

Split B holds out samples from parameter combinations already represented in training, isolating classification performance from generalisation, as described in Section 4.6.2. The best performances of each model in this split are presented in Table 5.3. The corresponding confusion matrices and per-class performance metrics for the CNN, XGBoost, and RF are shown in Figures 5.6, 5.8, and 5.9, respectively, while the CNN training curve is presented in Figure 5.7.

Table 5.3: Best-performing configuration per model on Split B.

Model	Stage	Imb. handling	Macro F1	Accuracy
CNN	Raw	None	90.19%	96.52%
XGBoost	Denoised	Augm	88.13%	95.95%
RF	Denoised	Augm	81.40%	91.48%



class	precision	recall	F1
new	92.9	83.0	**87.6**
used	97.1	98.9	**98.0**
worn	91.2	79.5	**84.9**
macro avg	**93.7**	**87.1**	**90.2**

Figure 5.6: Confusion matrix and per-class metrics for the CNN's best-performing configuration on Split B.

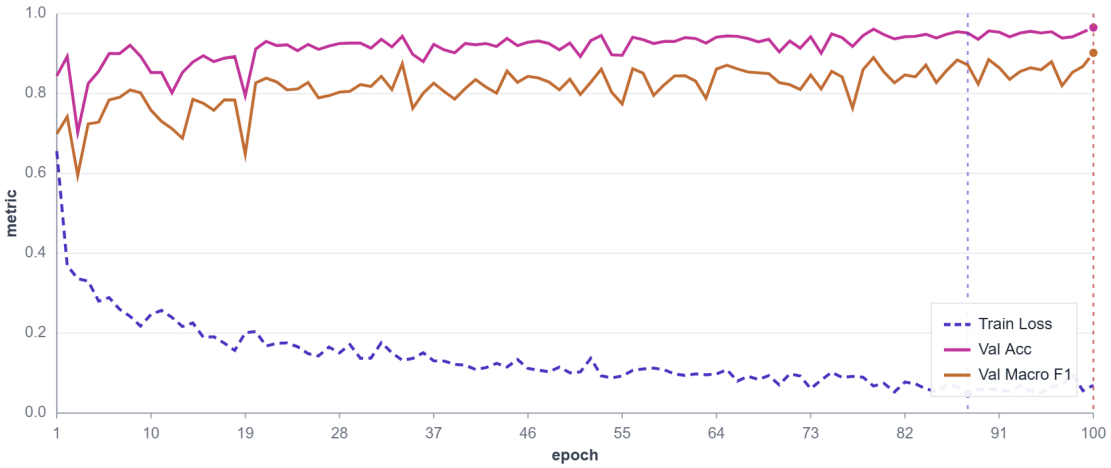
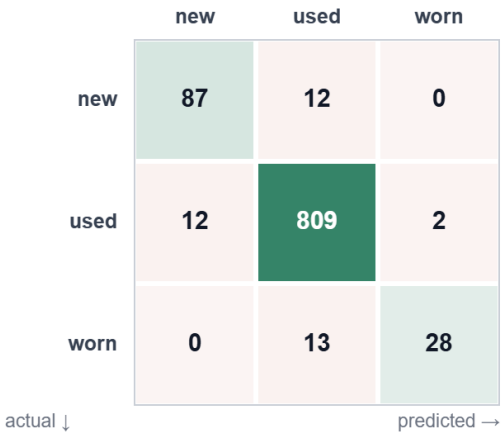
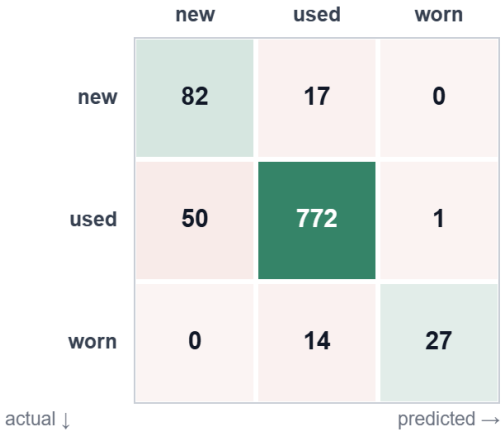


Figure 5.7: Training curve for the CNN's best-performing configuration on Split B.



class	precision	recall	F1
new	87.9	87.9	**87.9**
used	97.0	98.3	**97.6**
worn	93.3	68.3	**78.9**
macro avg	**92.7**	**84.8**	**88.1**

Figure 5.8: Confusion matrix and per-class metrics for the XGBoost best-performing configuration on Split B.



class	precision	recall	F1
new	62.1	82.8	**71.0**
used	96.1	93.8	**95.0**
worn	96.4	65.9	**78.3**
macro avg	**84.9**	**80.8**	**81.4**

Figure 5.9: Confusion matrix and per-class metrics for the random forest best-performing configuration on Split B.

On known parameters, the ranking inverts: the CNN leads with 90.19% macro F1, ahead of XGBoost's 88.13% and RF's 81.40%, and remains the only model whose best configuration favours the raw stage over the denoised stage. The per-class results show that the CNN's advantage comes mostly from the worn class, where it reaches an F1 of 84.9%, against 78.9% for XGBoost and 78.3% for RF. Its new-class F1 of 87.6% is close to XGBoost's 87.9% and well above RF's 71.0%, and its used-class F1 of 98.0% is the highest of the three, though only marginally ahead of XGBoost's 97.6%. Unlike on Split A, where the worn class was the CNN's clearest weakness, on Split B it becomes the class the CNN handles best relative to the two baselines. The training curve shows validation accuracy and macro F1 rising quickly within the first 10–20 epochs and then improving gradually, with far less fluctuation than on Split A, reaching their best values right at the final epoch: on known cutting conditions, additional training epochs keep translating into

better held-out results instead of overfitting. This improved performance is likely also due to the availability of more training data, which deep learning approaches require.

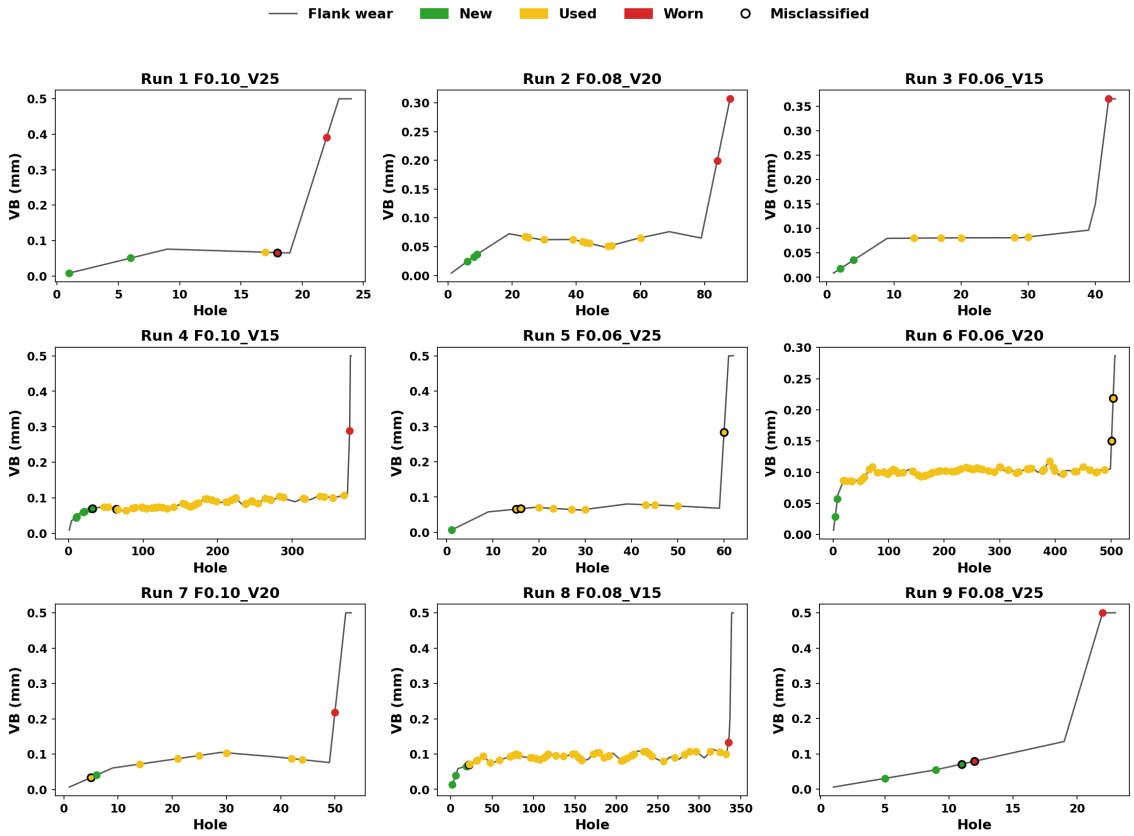


Figure 5.10: Real V_b vs predicted class in the best configuration of split B.

Figure 5.10 shows the same breakdown for the CNN on Split B, and the contrast with Split A is stark: misclassified holes are rare and sit almost exclusively near a threshold crossing, rather than spread across a plateau. This is consistent with the CNN's high aggregate accuracy on this split: once a cutting condition is partly represented in training, its errors shrink to the small set of inherently ambiguous labels rather than reflecting a broader failure to separate the classes.

5.2.1 Split comparison

Comparing Tables 5.2 and 5.3, all three models score noticeably higher on Split B than on Split A, as expected given that Split B tests interpolation within already-seen cutting conditions rather than generalisation to new ones. The size of that gap differs by model: macro F1 falls by 24.87% for XGBoost (88.13% to 63.26%), by 19.71% for RF (81.40% to 61.69%), and by 30.71% for the CNN (90.19% to 59.48%), making the CNN's split gap the largest of the three. Put simply, the classical models, for this data, extrapolate better while the CNN interpolates better : XGBoost and RF lead on Split A (63.26% and 61.69% macro-F1) but sit behind the CNN on Split B, whereas the CNN tops Split B (90.19%) but drops to third place on Split A (59.48%).

Breaking that gap down by class shows it is not spread evenly. The worn-class F1 falls by 27.4% for XGBoost (78.9% to 51.5%) and by 27.8% for RF (78.3% to 50.5%), but by 48.7% for the CNN (84.9% to 36.2%), nearly double the drop seen in the other two models. The new class also degrades from Split B to Split A for every model, though less consistently: XGBoost drops 43.1% (87.9% to 44.8%), the CNN drops 37.5% (87.6% to 50.1%), and RF drops 29.0% (71.0% to 42.0%). The used class is comparatively stable across the split for all three, with drops of only 2.5 to 5.8%.

Figures 5.5 and 5.10 make the performance gap between the two splits visible at the level of individual pecks: on Split B, nearly all of the CNN's errors cluster around the class boundaries. Runs like Run 5 and Run 4 are misclassified only where the real curve is actively crossing into the next class, while long, stable stretches within a class are essentially error-free. The exception is the runs with the fewest held-out pecks (Run 9, 6/15; Run 1, 3/13), whose error counts are inflated by the small-sample effect already discussed in Section 5.4, not by any different failure mode. Split A looks qualitatively different: XGBoost's misclassifications are spread across a held-out run's entire duration, not clustered at the transition. Run 6, for instance, is wrongly called *used* throughout most of its nearly 2 000 pecks in the *new* region, long before its real V_B approaches either threshold. This is a different kind of error than Split B's: it is not the model being unsure at an ambiguous margin, but the model having no reliable decision boundary at all for a cutting condition it never trained on, and defaulting broadly towards *used*, the majority training class. This adds a spatial reading to the class-level breakdown above: the worn class drives the largest share of the Split A gap in aggregate, but wherever a run is entirely novel, the resulting errors are not confined to the wear transition itself; they occur throughout the run, including in regions a well-generalising model should find easy.

This pattern is consistent with the CNN performing well when classifying wear states under cutting parameters it has already seen, but struggling on a condition entirely absent from training. This more plausibly reflects the amount of training data available for the unseen condition than a fixed limitation of the model: in Split A, each fold trains on only eight of the nine runs, so the held-out run's parameters are never observed during training, leaving the model no basis for placing its decision boundaries correctly in that region of the input space. The class breakdown makes this more specific than a blanket statement would suggest: the gap is driven overwhelmingly by the worn class, the one the CNN detects best on Split B, where the parameters are already known, and worst, by a wide margin, on Split A, where they are not.

5.3 Computational performance

This section reports the training and inference times measured for each model, broken down by validation split. Figure 5.11 shows the distribution of training times across all configurations, while Figure 5.12 presents the corresponding inference times.

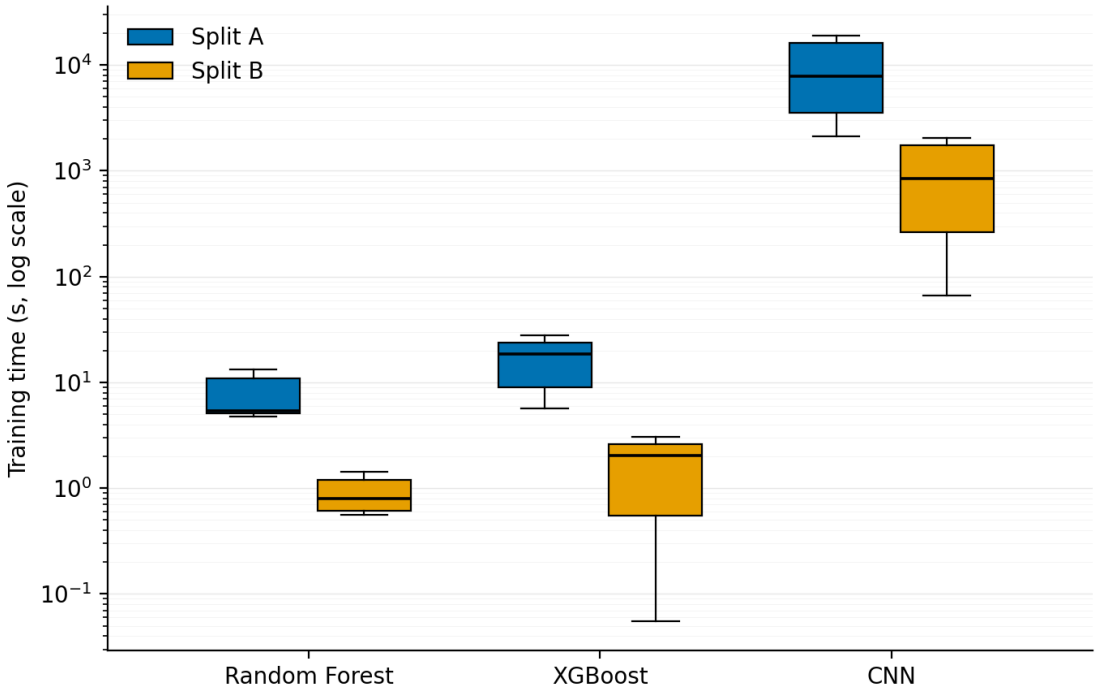


Figure 5.11: Training time across all configurations, by model and validation split.

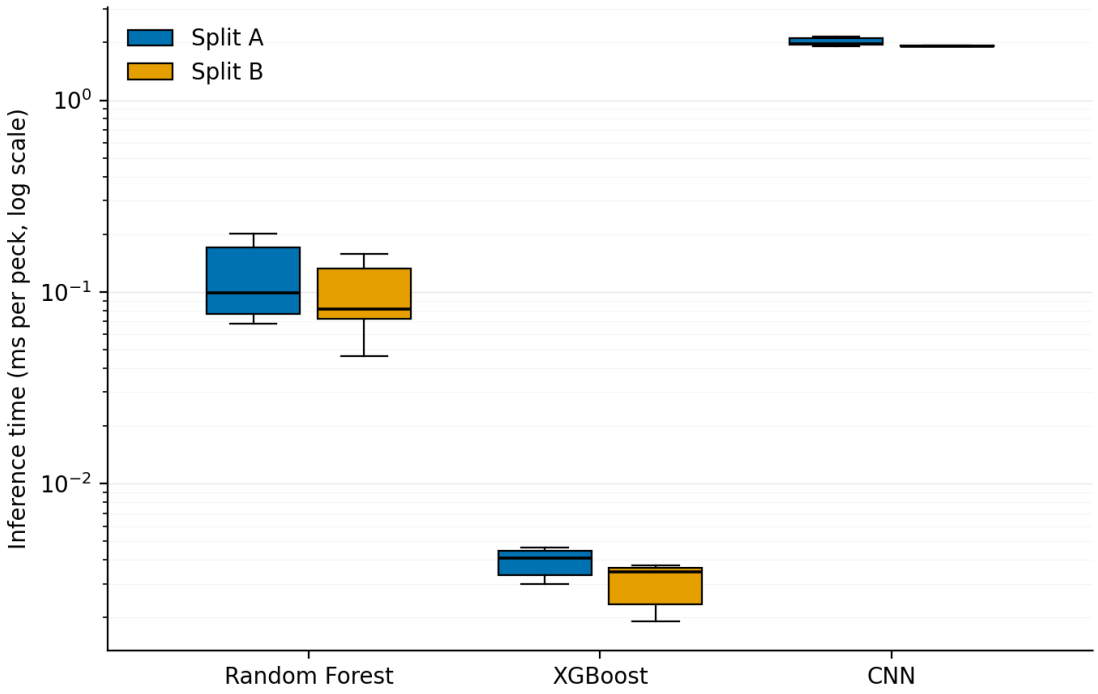


Figure 5.12: Inference time across all configurations, by model and validation split.

Split A

On Split A, each reported time is the sum of training across all 9 runs using LOOCV, since a separate model is fit per fold. At their best configurations, RF trains in 5.28 s and XGBoost in 25.33 s across the 9 folds combined, both negligible in absolute terms. Across the full experimental grid, though, this is not the whole picture: RF's training time ranges from 4.74 s to 13.24 s depending on stage, cap, and augmentation, and XGBoost's from 5.70 s to 27.78 s, so even the slowest classical configuration in the grid stays well under half a minute. The CNN's best Split A configuration trains in 1.86 hours (6 709.5 s), but the slowest configuration in the grid raw stage, full data, with augmentation takes 18.05 hours (64 990.0 s), nearly 10× longer than the best. This spread matters for anyone picking a configuration outside this thesis's specific best-performing one: a different choice of preprocessing or balancing strategy is not just a small accuracy trade-off; it can be almost a full day's difference in training time.

Inference time is far more stable across the grid than training time. XGBoost stays between 0.0030 and 0.0046 ms/sample, RF between 0.068 and 0.202 ms/sample, and the CNN between 1.921 and 2.149 ms/peck, regardless of configuration. Against a peck duration of roughly 2 to 3 s, even the CNN's slowest inference in the grid is under 0.11% of the drilling time itself; all three models are effectively negligible at inference, no matter which configuration is used.

This comparison is worth stating precisely, though, because "negligible inference time" does not mean real-time analysis in the strict sense. A peck's signal only becomes classifiable once the peck itself has finished cutting; the model needs the complete windowed signal, not a still-forming one, so a wear-class prediction for the peck currently being drilled is never available while it is still in progress. What the low inference times do provide is the next best thing: since classifying a peck takes a few milliseconds at most, against a peck that takes seconds to drill, the result for a just-finished peck is available essentially immediately once it ends, well before the next peck is underway. In practice this means the system reports the most recently completed peck's wear state with a lag of about one peck, not the current one, near-real-time monitoring rather than true real-time monitoring but close enough that the reported class is never more than a few seconds stale.

Split B

On Split B, each reported time is a single fit, since Split B trains one model on the pooled 85% of holes rather than 9 separate folds. RF trains in 1.43 s and XGBoost in 3.04 s at their best configurations. Across the full grid, RF ranges from 0.56 s to 1.43 s and XGBoost from 0.06 s to 3.04 s, both comfortably under a few seconds regardless of configuration. The CNN's best Split B configuration takes 0.75 hours (2 697.0 s), but the grid's slowest configuration reaches 1.81 hours (6 527.3 s), while its fastest, the cap-50 undersampled setting takes only 0.06 hours (210.5 s), around 30× faster than the slowest. As with Split A, the choice of configuration has a far bigger effect on CNN training time than on classical-model training time.

Inference times are essentially identical to Split A for every model: the CNN ranges from 1.907 to 1.936 ms/peck across the Split B grid, barely distinguishable from its Split A range, since inference cost is a property of the trained model's architecture rather than of which split produced it. The same near-real-time reasoning from Section 5.3 applies unchanged here.

5.3.1 Split comparison

Training time drops from Split A to Split B for every model at their best configurations, but by very different margins: 3.7× for RF (5.28 s to 1.43 s), 8.3× for XGBoost (25.33 s to 3.04 s), and 2.5× for the CNN (1.86 h to 0.75 h). Most of this gap is structural, not a property of the models themselves: Split A sums training across 9 leave-one-file-out folds, each trained on close to the full dataset, while Split B fits once on the pooled training holes. For a model whose per-fit cost scales roughly linearly with data volume, summing 9 folds this size should cost close to 9× a single Split B fit almost exactly what XGBoost shows (8.3×). RF's smaller ratio (3.7×) suggests a larger fixed overhead per fit that dilutes the effect of fold count. The CNN's ratio is smaller still (2.5×) for a different reason: its best Split A configuration uses a 50-hole undersampling cap, so each of its 9 folds trains on a much smaller budget than Split B's full pooled training set; the fold-count effect and the data-volume reduction partly cancel out. The same structural gap shows up across the full grid, not only at the best configurations: the CNN's slowest Split A run (18.05 h) is still roughly 9.9× its slowest Split B run (1.81 h).

Inference time, by contrast, does not meaningfully differ between splits for any model, since it depends on the trained model's architecture and input size, not on how the training/test partition was constructed. The practical takeaway holds regardless of split and regardless of configuration: RF and XGBoost are cheap enough for frequent retraining under any setting tested, and all three models classify a peck fast enough, relative to how long a peck takes to drill, to support near-real-time one-peck-lagged, not instantaneous monitoring on the shop floor.

5.4 Per-run performance variance

A key caveat of the results shown above is that they depend heavily on which run is held out. Table 5.1 reports one macro-F1 per model, averaged across nine different held-out runs, but that average can mask large per-run differences: a model could score well on most runs and still fail badly on two or three specific ones, and the aggregate number alone would not reveal which. This section examines that spread directly, for both Split A and Split B.

Split A

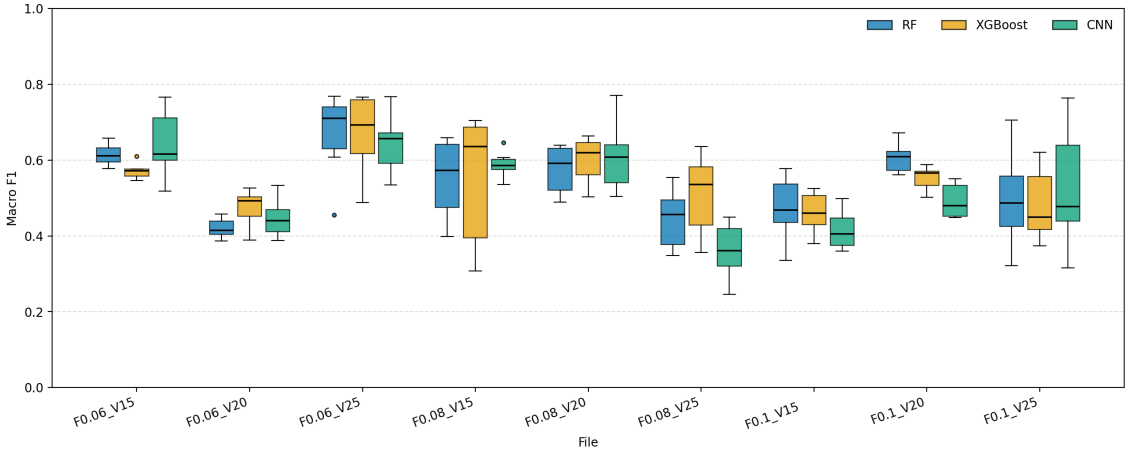


Figure 5.13: Per-run macro F1 across the six main-grid configurations, Split A.

Figure 5.13 shows this per-run spread directly, averaged over the six main-grid configurations. XGBoost is the tightest of the three here, with a macro-F1 coefficient of variation of 12.47% (standard deviation 6.74); RF is somewhat more scattered at 15.63% (standard deviation 8.42). The CNN remains the widest of the three, with a standard deviation of 10.31 and a coefficient of variation of 19.64%, and its balanced-accuracy CV (15.70%) tells the same story. Pooled across all 27 per-run averages (9 runs $\times$ 3 models), macro-F1 averages 53.50% with a standard deviation of 8.31 (CV 15.54%) and ranges from 36.06% to 66.87%. Unlike the single-configuration comparison, the two ends of that range do not belong to the same model: the CNN produces the single worst per-run result (Run 9), while XGBoost produces the single best (Run 5), with RF close behind on the same run. All three models reach their individual peak on Run 5 (F0.06_V20), but they diverge sharply on Run 9.

Table 5.4: Split A per-run macro F1 by model, averaged across the six main-grid configurations.

Run	n	RF	XGBoost	CNN	Mean	Std
Run 1	90	49.78%	48.21%	52.54%	50.18%	2.19
Run 2	353	57.53%	60.02%	60.97%	59.51%	1.78
Run 3	200	61.43%	57.18%	64.21%	60.94%	3.54
Run 4	1511	47.24%	46.09%	41.60%	44.98%	2.98
Run 5	248	66.62%	66.87%	64.33%	65.94%	1.40
Run 6	2031	42.01%	47.41%	44.68%	44.70%	2.70
Run 7	210	60.63%	55.29%	49.19%	55.04%	5.72
Run 8	1358	55.15%	55.00%	58.86%	56.34%	2.19
Run 9	92	44.59%	50.89%	36.06%	43.85%	7.44

n is the peck count of the held-out file, as used by RF and XGBoost. The CNN windows this into 2.0 s segments before classification, so its effective per-fold sample count is slightly lower negligible for the longer runs, but noticeably so for the three shortest (Run 1: 68, Run 3: 167, Run 9: 68).

From Table 5.4 it can be seen that Run 6, the largest held-out file at 2031 pecks, is one of the two hardest runs on average (44.70% mean macro-F1) consistent with the idea that its slow wear progression, and correspondingly long, hard-to-separate transition through the healthy class, makes it a genuinely difficult condition. Run 9 is the other hardest run (43.85%), but for a different reason: at only 92 held-out pecks, it is one of the two smallest files in the dataset, and its combination of the lowest mean and by far the largest cross-model disagreement (std 7.44) points to a small-sample effect rather than a shared difficulty. Neither run is uniformly the hardest fold for every model: Run 6 is RF's worst fold (42.01%) but not XGBoost's or the CNN's, while Run 9 is the CNN's worst fold by a wide margin (36.06%), yet XGBoost handles it reasonably (50.89%), its best relative showing of the three models on that run.

The standard deviation across models, in the last column, again splits the runs into two groups. Run 5, Run 2, and Run 1 show the closest agreement between models, with a std of 1.40, 1.78, and 2.19, so whatever makes these conditions easy or hard looks like a property of the condition itself rather than of which model is tested on it. Run 9 is the clearest outlier in the opposite direction, with a std of 7.44, driven by the CNN's especially weak result there; Run 7 stands out too, where the CNN (49.19%) trails both RF (60.63%) and XGBoost (55.29%) by a wide margin.

Taken together, these results still say something the aggregate numbers in Table 5.1 do not: performance is uneven across the operating envelope, and a handful of individual runs diverge sharply from each model's own average. The CNN remains the most volatile of the three, but its extremes are less clear-cut than they first appear: on its own best run, Run 5, it reaches 64.33%, actually the lowest of the three models on that run, since RF and XGBoost peak on the same run and edge slightly higher (66.62% and 66.87%). On its worst run, Run 9, it drops to 36.06%, well below RF's and XGBoost's own worst runs (42.01% and 46.09%). Even so, the CNN's spread from best to worst run, 28.27%, is still the widest of the three models (24.61% for RF, 20.78% for XGBoost). Any claim about which model generalises best still needs that qualification. On

several individual cutting conditions, the ranking between models looks quite different from what the headline numbers in Table 5.2 suggest.

Split B

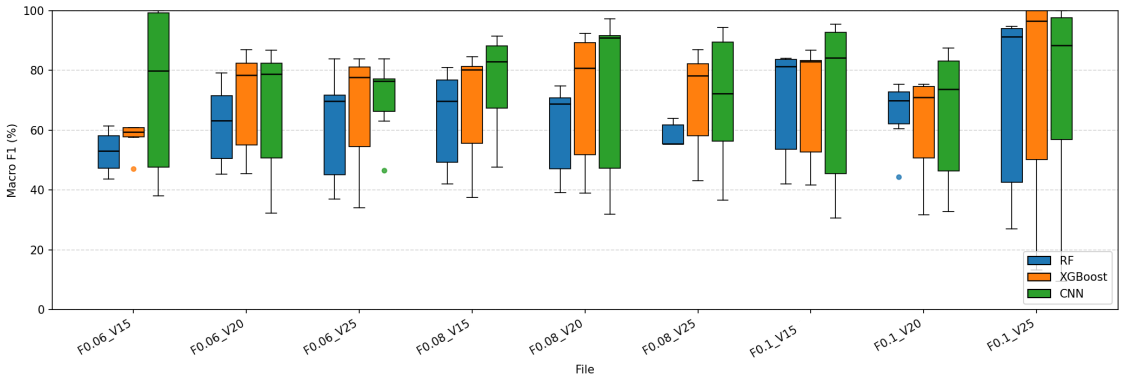


Figure 5.14: Per-run macro F1 across the six main-grid configurations, Split B.

Figure 5.14 and Table 5.5 break down performance by run for Split B, the same way Section 5.4 did for Split A but the *n* column here means something different. Split B's test set is a stratified 15% sample of holes drawn from within every run, pooled together into one test set; the *n* for a given run is only that run's slice of the pooled set (roughly 15% of its holes), not the whole run. The other 85% of that run's pecks were in training, mixed with training data from all 8 other runs. This is unlike Split A, where a held-out run's macro-F1 is computed on the entire run because none of it was seen during training. That difference matters here because several of these slices are tiny: as few as 13 pecks (Run 1, CNN) up to 312 (Run 6). A 15-30 peck slice can move several points of macro-F1 on the outcome of one or two misclassified holes, in a way a 150+-peck slice cannot. So the per-run spread below reflects a mix of genuine per-condition difficulty and small-sample noise, unlike Split A's, where it reflects generalisation difficulty on distinct, unseen cutting conditions alone.[1]

Each model's spread between its best and worst run, averaged across the six main-grid configurations (Table 5.5), is itself informative. RF swings by 18.18% (70.94% on Run 1 down to 52.76% on Run 3), XGBoost by 16.13% (73.76% down to 57.63%, on the same two extremes as RF: Run 1 is both models' best run, Run 3 both models' worst), and the CNN by only 10.92% (76.19% on Run 8 down to 65.27% on Run 7). The CNN is again the most consistent of the three across cutting conditions within this split (coefficient of variation 4.67%, against 8.89% for RF and 7.53% for XGBoost), which fits the extrapolation/interpolation framing already established for this chapter:

[1] The CNN numbers in this section come from a separate training run from the one behind Table 5.1: the CNN is not seeded and its weights are not persisted, so no two training runs reproduce the same per-fold result even for an identical configuration. The aggregate macro-F1 values here therefore differ from the table by a few points, though the qualitative pattern the CNN being the most consistent of the three models holds across runs.

whatever the CNN is learning generalises more evenly across the pooled, in-distribution test than either handcrafted-feature model does.

Table 5.5: Split B per-run macro F1 by model, averaged across the six main-grid configurations.

Run	n	RF	XGBoost	CNN	Mean	Std
Run 1	17	70.94%	73.76%	72.27%	72.32%	1.41
Run 2	60	60.52%	71.24%	72.66%	68.14%	6.64
Run 3	34	52.76%	57.63%	73.44%	61.28%	10.81
Run 4	232	69.62%	70.13%	70.47%	70.07%	0.43
Run 5	44	61.65%	67.13%	70.63%	66.47%	4.53
Run 6	312	61.98%	70.07%	66.91%	66.32%	4.08
Run 7	36	65.52%	61.51%	65.27%	64.10%	2.25
Run 8	208	64.02%	68.65%	76.19%	69.62%	6.14
Run 9	20	58.35%	70.33%	70.32%	66.33%	6.91

n is the size of that run's slice of the pooled 15% test set, as used by RF and XGBoost. The CNN windows this into 2.0 s segments before classification, so its effective slice is slightly smaller for the three shortest runs (Run 1: 13, Run 3: 29, Run 9: 15).

No single run is uniformly hardest across all three models. Run 3, with only 34 test pecks for RF/XGBoost and 29 for the CNN, is both classical models' worst run, but it is the CNN's best; the resulting std of 10.81 the largest in the table is driven entirely by the CNN handling this run well while the other two do not. Run 7 shows the opposite asymmetry: it is the CNN's worst run, but only middling for RF and XGBoost. Read alongside Split A, where every run gets a full, unseen test fold, this model-dependence is consistent with Split B measuring something closer to noise on individual runs than genuine per-condition difficulty.

5.4.1 Split comparison

Comparing the spread between each model's best and worst run across the two splits gives a first read on which split's per-run variance is larger. For this analysis, Table 5.6 was created.

Table 5.6: Per-run macro-F1 spread and standard deviation, Split A vs Split B.

Model	Split A spread	Split A std	Split B spread	Split B std
RF	24.61%	8.43%	18.18%	5.58%
XGBoost	20.78%	6.74%	16.13%	5.11%
CNN	28.27%	10.31%	10.92%	3.31%

All three models show less per-run spread on Split B than on Split A, consistent with Split B being the easier, in-distribution test: whichever run you pick, there is simply less room for a model to fail badly when the test pecks come from a condition it has partly trained on. The effect

size differs by model, though. The CNN's spread shrinks the most in relative terms (28.27% down to 10.92%, well under half), while the two classical models narrow less (RF: 24.61% to 18.18%; XGBoost: 20.78% to 16.13%). This is the same pattern already established in Sections 5.4 and 5.4: the CNN is the most volatile model under extrapolation to unseen conditions, but the most stable of the three once it has partial exposure to the condition being tested.

The more informative comparison is whether the same runs are hard across both splits. Looking at individual runs, they largely are not. Run 9 is the clearest illustration for the CNN: it is the model's worst run on Split A by a wide margin (36.06%, its only score below 40%), yet on Split B it lands at 70.32%, an entirely unremarkable result within a point of the model's own Split B average (70.91%). The same cutting condition, the same model, one appears catastrophic and the other looks routine, depending on whether the run is held out entirely or only partly sampled. XGBoost shows a version of the same thing on Run 1: one of its three weakest runs on Split A (48.21%, against a model mean of 54.11%), yet its single best run on Split B (73.76%).

This is the central takeaway of the per-run comparison: Split A's per-run variance and Split B's per-run variance are not two views of the same underlying difficulty, and a run being hard to generalise to (Split A) says little about whether that run's own held-back holes are hard to classify (Split B). This is exactly what the two splits are designed to measure separately. Split A is about extrapolation to conditions the model has never seen at all; Split B is about interpolation within conditions it has partly seen, and cases like Run 9 for the CNN and Run 1 for XGBoost suggest the splits are actually probing different failure modes, not redundant versions of the same test. Taken together with the aggregate results in Section 5.1, no single cutting condition can be flagged as a universal hard case for this task; difficulty is split-dependent, and how a model handles a given condition under extrapolation says little about how it handles the same condition in-distribution.

5.5 Ablation study

To isolate the contribution of each design choice, a systematic ablation study was conducted by varying one factor at a time while keeping the remaining parameters fixed. The study is presented for Split A first and then for Split B, since Split A represents the more realistic and more challenging evaluation scenario, testing generalisation to cutting conditions unseen during training rather than performance within known ones.

Each ablation below uses every model's best-performing configuration for the split under discussion as its base (Tables 5.2 and 5.3), varying only the axis being examined.

5.5.1 Effect of peck selection

Split A

In peck drilling, each hole is produced through a sequence of successive peck cycles. This ablation compares the use of different peck positions, restricting the training and test data to a single peck instead of pooling all four.

Table 5.7: Effect of peck selection on Split A macro-F1 and accuracy, at each model's best configuration.

Model	Peck	Macro F1	Accuracy
RF	1	63.21%	90.20%
RF	2	62.61%	88.79%
RF	All	61.69%	86.74%
RF	4	60.43%	88.60%
RF	3	59.01%	88.23%
XGBoost	2	68.26%	89.98%
XGBoost	3	67.18%	89.55%
XGBoost	1	66.39%	89.48%
XGBoost	All	63.26%	88.28%
XGBoost	4	59.94%	85.55%
CNN	All	59.48%	86.11%
CNN	4	56.49%	82.89%
CNN	2	53.02%	71.50%
CNN	3	52.53%	63.69%
CNN	1[a]	33.12%	52.51%

[a]*Computed over 8 of the 9 LOOCV folds: Run 9 has no 2.0 s windows at all for peck 1 (its peck-1 signal is shorter than the window length), so that fold was excluded rather than forced.*

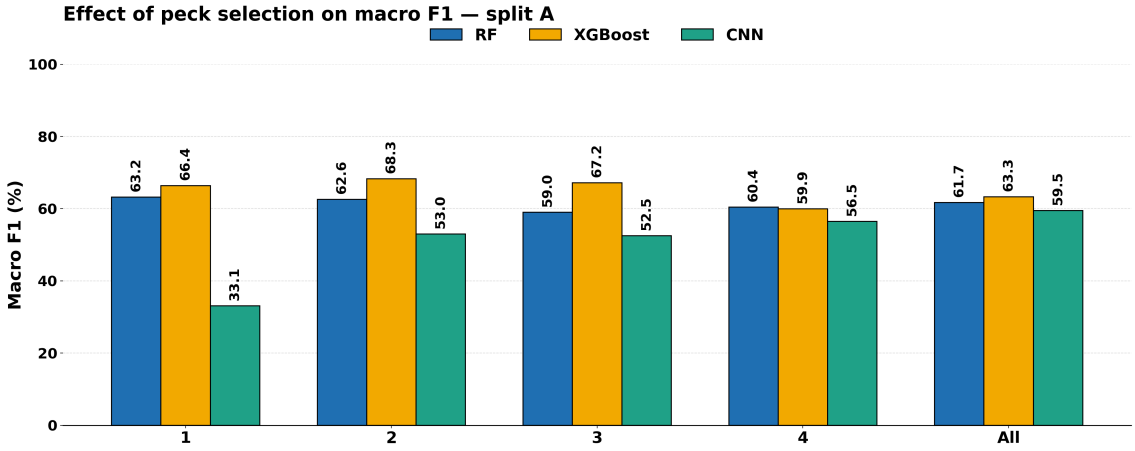


Figure 5.15: Effect of peck selection on Split A macro F1, at each model's best configuration.

Table 5.7 and Figure 5.15 show that peck 1 is the CNN's weakest individual peck by a wide margin (33.12%, partly reflecting the excluded fold above), well below pecks 2-4 (52-56%) and the weakest result in the whole table. Drilling is steady-state throughout, so this isn't a transient-regime effect. The more likely explanation is that no centre (pilot) hole was used, so the bit is unguided at entry on the first peck, adding vibration unrelated to wear that plausibly degrades the

signal the model relies on. Pooling all four pecks still beats every individual peck for the CNN (59.48%), reinforcing that it depends on the larger pooled sample more than the classical models, whose handcrafted features already summarise a whole peck; RF and XGBoost are unaffected by the window change, and peck 2 remains their strongest individual peck.

Training on a single peck still cuts CNN training time substantially (about 30 minutes per peck versus roughly 1.9 hours pooled, a 73% reduction), but this doesn't change the recommendation to pool all pecks, since real drilling jobs don't reliably indicate peck count in advance.

Split B

Table 5.8: Effect of peck selection on Split B macro-F1 and accuracy, at each model's best configuration.

Model	Peck	Macro F1	Accuracy
CNN	All	90.19%	96.52%
XGBoost	4	88.54%	96.67%
XGBoost	All	88.13%	95.95%
XGBoost	3	85.91%	94.61%
CNN	3	85.78%	93.36%
RF	3	84.71%	92.95%
CNN	1	84.60%	95.26%
XGBoost	2	83.56%	94.61%
RF	4	82.32%	93.75%
CNN	2	81.56%	93.78%
RF	All	81.40%	91.48%
XGBoost	1	79.73%	92.95%
CNN	4	79.44%	91.67%
RF	2	78.35%	92.12%
RF	1	77.68%	92.95%

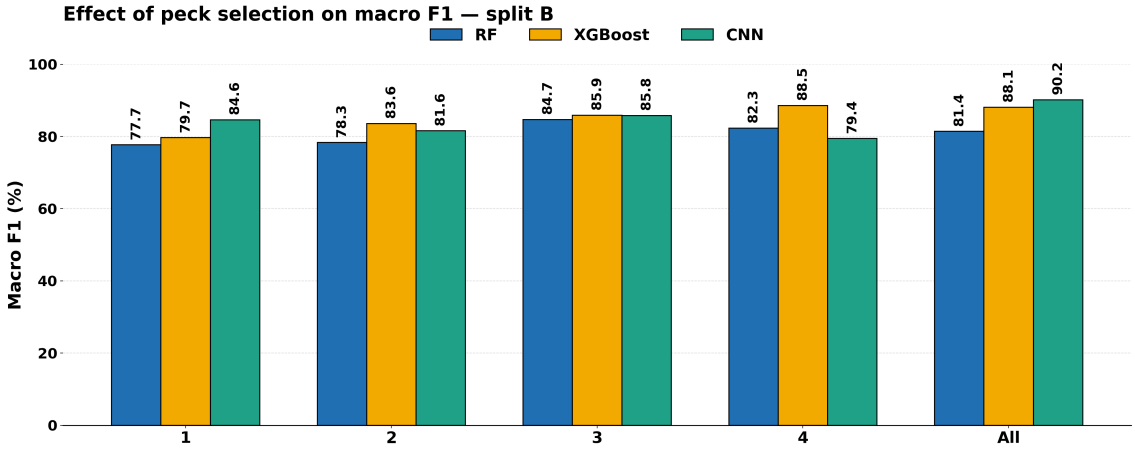


Figure 5.16: Effect of peck selection on Split B macro F1, at each model's best configuration.

Table 5.8 and Figure 5.16 show that, unlike Split A, pooling all pecks stays necessary only for the CNN. XGBoost's peck 4 alone (88.54%) edges out its pooled result (88.13%), and RF's peck 3 (84.71%) and peck 4 (82.32%) both beat its pooled result (81.40%): the classical models' handcrafted features don't need every peck pooled on this easier, in-distribution test, while the CNN still benefits from the larger pooled sample, consistent with its Split A behaviour.

5.5.2 Effect of channel set

Split A

Six channel configurations were evaluated: cutting forces (Fx, Fy, Fz), spindle current, three-axis accelerometer, microphone, and the combined multi-branch feature set. This ablation isolates the contribution of each sensor group by running a single channel group at a time, holding peck selection at all pecks.

Table 5.9: Effect of channel set on Split A macro-F1 and accuracy, at each model's best configuration

Model	Channel set	Macro F1	Accuracy
RF	All	61.69%	86.74%
RF	Forces	51.85%	82.88%
RF	Audio	48.98%	81.16%
RF	Accel	43.25%	78.55%
RF	Current	38.86%	59.53%
XGBoost	All	63.26%	88.28%
XGBoost	Forces	51.44%	82.59%
XGBoost	Audio	49.04%	86.12%
XGBoost	Accel	46.04%	82.37%
XGBoost	Current	39.99%	58.31%
CNN	All	59.48%	86.11%
CNN	Audio	43.32%	73.42%
CNN	Current	38.81%	56.35%
CNN	Accel	36.98%	56.64%
CNN	Forces	33.97%	53.84%

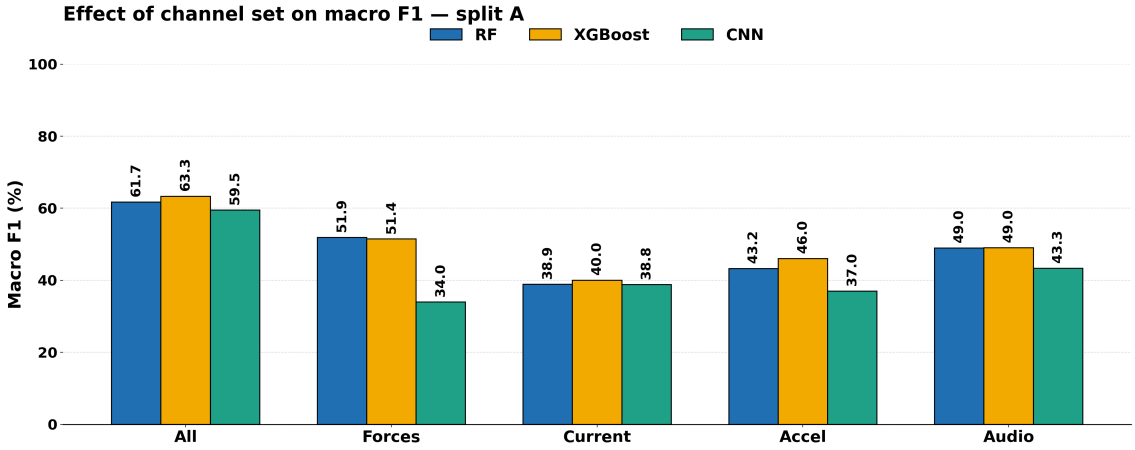


Figure 5.17: Effect of channel set on Split A macro F1, at each model's best configuration.

Table 5.9 and Figure 5.17 show that, for every model, full sensor fusion beats any single channel, confirming the value of sensor fusion for this task. At the 2.0 s window, the CNN's best single channel is audio (43.32%, up from spindle current's 46.02% at 1.0 s), with current second (38.81%) and forces still the CNN's weakest (33.97%) the same divergence as before: forces remains RF's and XGBoost's strongest single channel and the CNN's weakest, consistent with the CNN extracting less from the raw force waveform than handcrafted features do.

Dropping to a single channel cuts CNN training time by about 93% (forces-only: ∼0.13 h vs 1.86 h for full fusion), similar in magnitude to before.

Split B

Table 5.10: Effect of channel set on Split B macro-F1 and accuracy, at each model's best configuration.

Model	Channel set	Macro F1	Accuracy
CNN	All	90.19%	96.52%
XGBoost	All	88.13%	95.95%
XGBoost	Forces	83.31%	94.18%
RF	All	81.40%	91.48%
XGBoost	Current	78.40%	92.00%
RF	Forces	76.38%	88.79%
RF	Current	74.71%	87.64%
CNN	Current	74.14%	87.04%
CNN	Forces	67.80%	89.25%
CNN	Accel	65.44%	89.15%
RF	Accel	62.63%	86.19%
XGBoost	Accel	61.96%	88.37%
XGBoost	Audio	55.45%	86.19%
RF	Audio	53.55%	79.34%
CNN	Audio	44.80%	83.35%

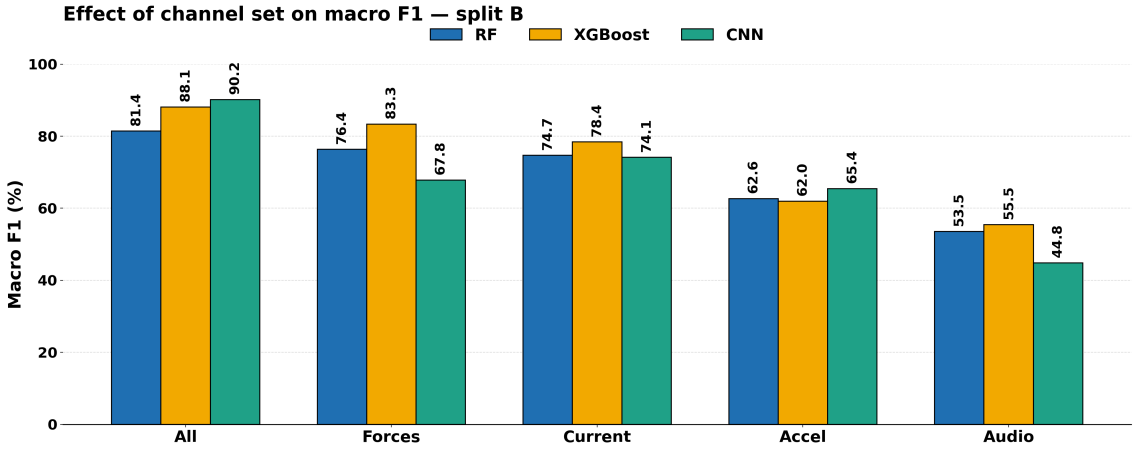


Figure 5.18: Effect of channel set on Split B macro F1, at each model's best configuration.

Table 5.10 and Figure 5.18 show that full sensor fusion is still best for every model, as on Split A. The individual-channel ranking shifts, though: forces becomes XGBoost's (83.31%) and RF's

(76.38%) strongest single channel, while remaining the CNN's weakest besides audio (67.80%) the same forces/CNN mismatch already noted on Split A. Audio is the weakest single channel for every model here, including the CNN (44.80%, close to its Split A floor).

5.5.3 Effect of window size

Split A

This ablation is CNN-only, since window size does not apply to the classical, feature-based path. The three best-performing CNN configurations on Split A were re-run with the input window length halved from 2.0 s to 1.0 s, keeping the 25% overlap and all other settings fixed.

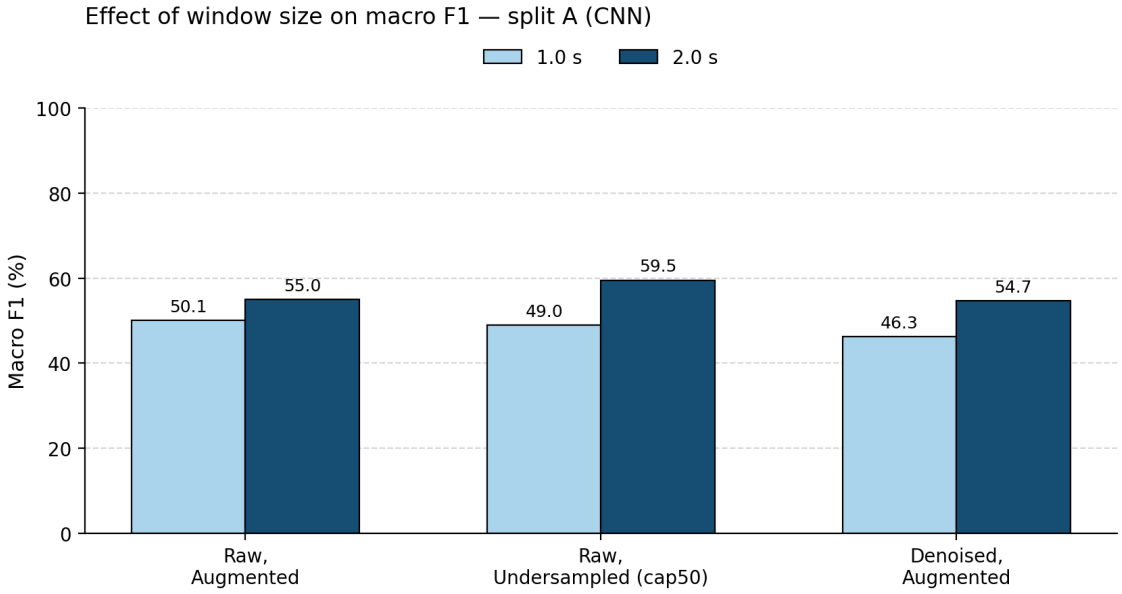


Figure 5.19: Effect of window size on Split A macro F1, for the top-3 CNN configurations.

As Figure 5.19 shows, using 2.0 s instead of 1.0 s windows improves all three configurations tested: +4.93% (augmented, raw stage), +10.43% (cap-50), and +8.37% (denoised stage) macro-F1. A longer window likely captures more steady-state cutting signal per sample, though this is an interpretation rather than a confirmed cause. The fusion gap noted above also widens at the longer window, from 4.05% at 1.0 s to 16.16% at 2.0 s, meaning sensor fusion is doing comparatively more work as the window grows. The window-size gain is also large enough to reorder the three configurations tested, and the cap-50 configuration overtakes the raw-stage configuration, becoming the best-performing configuration.

Split B

This axis is CNN-only. Three configurations were re-run at 1.0 s and 2.0 s, holding everything else fixed.

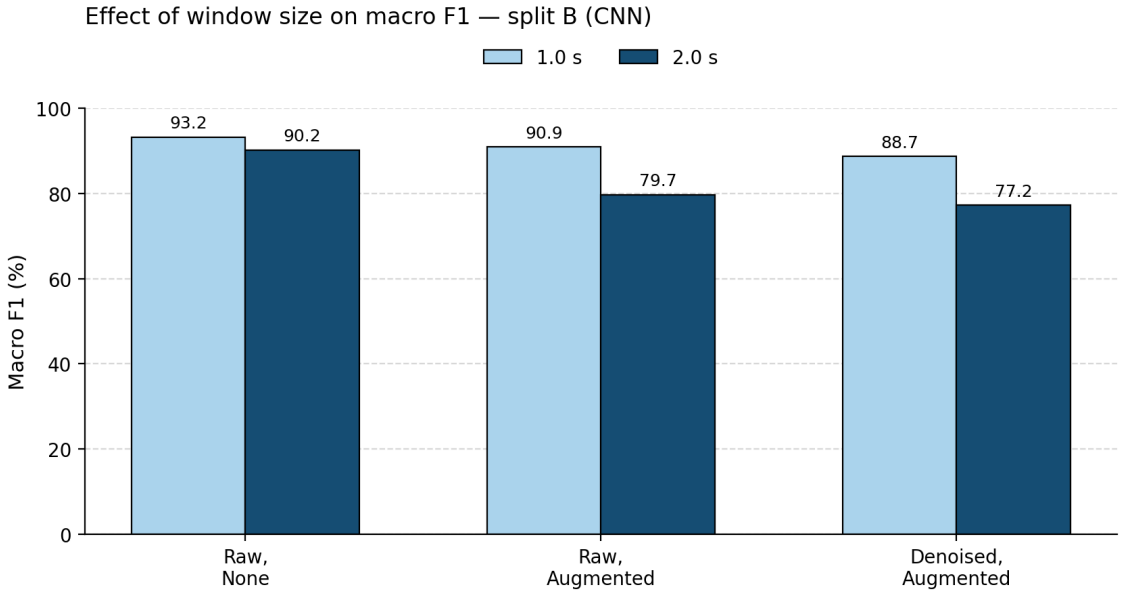


Figure 5.20: Effect of window size on Split B macro F1, for three CNN configurations (1.0 s vs 2.0 s).

Figure 5.20 shows the one axis where Split B disagrees with Split A. The reported configuration (raw/none) loses 3.01% going from 1.0 s to 2.0 s, and the two augmented configurations lose considerably more (11.19% and 11.57%). Given how few held-out pecks some Split B folds carry (Section 5.4), this reversal is worth noting but should not be read as a settled explanation of window size's effect in general.

5.5.4 Split comparison

Comparing the ablation axes across both splits shows two patterns: some findings hold regardless of split, and one inverts completely.

Peck selection is the most stable axis (Tables 5.7 and 5.8): pooling all four pecks is necessary for the CNN on both splits, but merely convenient for RF and XGBoost, which can each be edged out by a single well-chosen peck (peck 1/2 on Split A, peck 3/4 on Split B). Since which peck wins isn't stable, pooling all pecks remains the practical recommendation either way.

Channel set is stable in ranking but not in magnitude (Tables 5.9 and 5.10): full fusion wins for every model on both splits, and forces is the classical models' strongest single channel and the CNN's weakest on both. The fusion advantage shrinks for the classical models on the easier Split B (RF: 9.84% $\rightarrow$ 5.02%; XGBoost: 11.82% $\rightarrow$ 4.82%) but stays flat for the CNN (16.16% vs 16.05%) in-distribution data lets RF and XGBoost close most of the gap to fusion with their single best channel; the CNN gets no comparable benefit.

Class balancing (Table 5.1) is where the splits disagree most sharply, and only for the CNN. RF and XGBoost are consistent on both splits: undersampling is always the clear loser, and augmentation edges out no balancing by a small margin either way. The CNN inverts completely its

best strategy on Split A (undersampling) is its worst on Split B, while no balancing, merely middling on Split A, becomes clearly best on Split B. Nothing about the classical models' balancing behaviour predicts the CNN's, and the CNN's own behaviour does not carry from one split to the other.

Window size inverts completely (Figures 5.19 and 5.20): every configuration gains from doubling the window on Split A (+4.93 to +10.43%) and loses from the same change on Split B (-3.01 to -11.57%). A longer window plausibly trades more context per sample (which may help more on Split A, where generalising to unseen conditions is the challenge) for fewer non-overlapping training windows per peck (which may hurt more on Split B, where the conditions are already in-distribution and training volume is the more likely bottleneck). This is a plausible reading of a consistent pattern, not a confirmed mechanism, and Split B's smaller per-fold sample sizes mean it should be treated as tentative.

Table 5.11: Range between the CNN's best and worst results on each ablation axis, by split.

Axis	Split A range	Split B range
Peck selection	26.36%	10.75%
Channel set	25.51%	45.39%
Class balancing	4.48%	41.57%
Window size (best config)	10.43% gain	3.01% loss

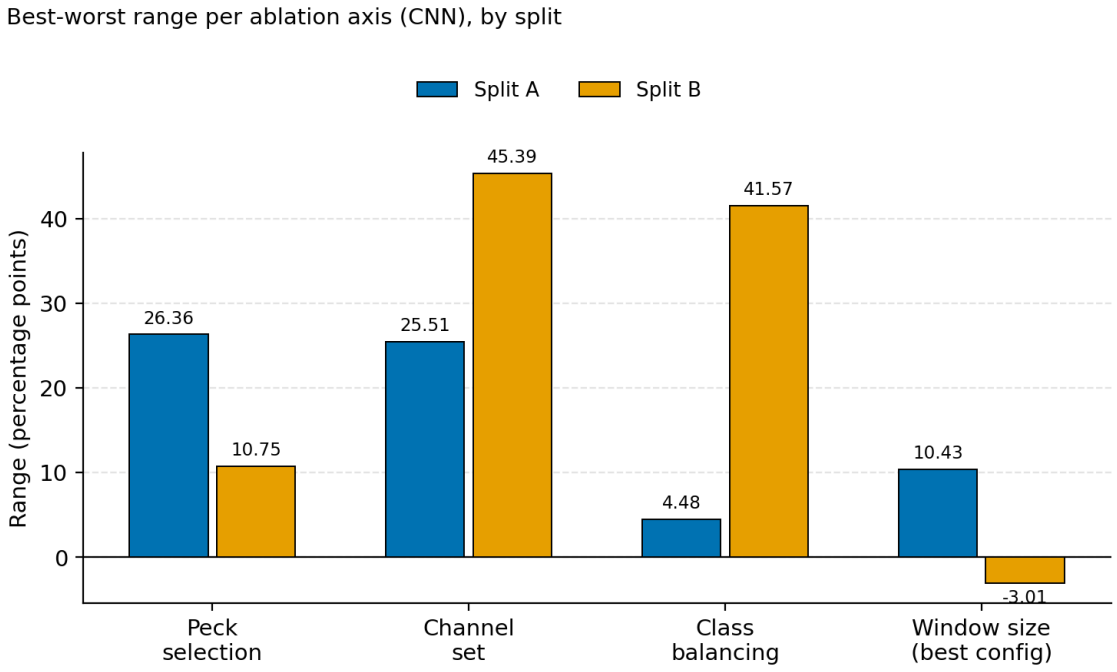


Figure 5.21: Best-worst range for the CNN on each ablation axis, by split.

Table 5.11 and Figure 5.21 summarise this: which design choice matters most for the CNN depends on the split. On Split A, peck selection and channel set carry the highest stakes (∼26%

each), while class balancing is nearly free (4.48%). On Split B, that ranking flips: class balancing and channel set become the highest-stakes choices (41-45% each), while peck selection matters comparatively little (10.75%). A choice that looks safe to get roughly right under one evaluation protocol can be the largest source of error under the other an argument for reporting both splits' ablations rather than assuming conclusions transfer from one to the other.

Chapter 6

Conclusions and future perspectives

6.1 Summary of main findings

This dissertation set out to build and test a multi-sensor TCM system for drilling AISI 316L stainless steel, pitting a classical machine learning baseline (RF and XGBoost) against a multi-branch one-dimensional CNN that works directly on raw sensor signals.

Which model came out on top depended entirely on the validation split. On Split B, where training and test data come from the same cutting conditions, the CNN was the best performer, reaching 90.19% macro-F1 against 88.13% for XGBoost and 81.40% for RF. Move to Split A, where each fold tests a cutting condition the model has never seen, and the picture flips: XGBoost takes the lead at 63.26%, RF sits close behind at 61.69%, and the CNN drops to 59.48%, third place but by less than 2.5 points. The CNN also has by far the largest gap between the two splits: 30.71 percentage points in macro-F1, versus roughly 20 for the tree-based models. Taken together, this still suggests the CNN fits the training-parameter distribution rather than learning the wear signature itself. This part should actually transfer to new cutting conditions.

This may also be a data problem rather than a purely architectural one: deep networks typically need substantially more training examples than tree-based models to learn representations that generalise, and Split A trains the CNN on only eight files per fold, a constraint that does not equally bind RF or XGBoost's much lower-dimensional feature space.

Looking fold by fold, averaged across the six main-grid configurations (Tables 5.4 and 5.5), the two splits tell different stories. On Split A, no single run is hardest for every model, but Run 4 and Run 6 are consistently among the three worst runs for all three models, while Run 5 is each model's single best run part of the difficulty here tracks the cutting condition itself, not just the model. On Split B, that pattern disappears entirely: no run is uniformly hard or easy across models. Run 3, for example, is both classical models' worst run but the CNN's best, and no run ranks first for all three. The CNN is also the least predictable of the three on Split A, matching or beating the classical models on its good folds but falling well short on its bad ones, while on Split B it is the most consistent of the three across cutting conditions. As Section 5.4.1 established, a run being hard to generalise to on Split A says little about whether that run's own held-back holes

are hard to classify on Split B the two splits probe different failure modes, not two views of the same difficulty.

A few things from the ablation study are worth keeping in mind. Which peck carried the most information depended on the model: for RF, peck 1 alone slightly outperformed pooling all four; for XGBoost, peck 2 did; for the CNN, no individual peck matched pooling all four, which still gave its best Split A result. Pooling remained the right call for the final pipeline regardless, since real drilling jobs won't always tell you how many pecks to expect. Sensor fusion paid off everywhere: no single channel group ever beat the full set, for any model. What differed was which channel mattered most on its own: forces for RF and XGBoost, audio for the CNN, with forces the CNN's weakest channel of all, which suggests the CNN isn't extracting as much from the raw force waveform as the handcrafted features manage to pull out of it. Class-imbalance handling was the least consistent axis: undersampling clearly hurt RF and XGBoost, but was actually the CNN's best strategy on Split A, only to become its worst on Split B, while no balancing and augmentation stayed close competitors for the classical models on both splits. And doubling the CNN's input window from 1.0 s to 2.0 s helped consistently on Split A but hurt just as consistently on Split B, the one ablation axis where the two splits disagreed outright.

None of this is close in practice. XGBoost and RF train in seconds and classify a peck in under a millisecond, on a CPU. The CNN needs a GPU and, in its slower configurations, the better part of a day to train, even though its own inference time is still negligible next to how long a peck takes to drill. For the conditions tested here, XGBoost is the more practical choice; the CNN only really earns its keep within the range of cutting conditions it was trained on.

6.2 Contributions of the present work

Beyond the specific numerical results, this work contributes the following to the field:

- A labelled, multi-sensor drilling dataset for AISI 316L stainless steel spanning a full-factorial grid of nine cutting conditions
- A documented, reproducible signal-processing pipeline and its performance
- A controlled, direct comparison between classical handcrafted-feature models (Random Forest, XGBoost) and a raw-signal multi-branch 1D-CNN trained and evaluated on identical data, splits, and imbalance-handling strategies
- A dual-split validation methodology run-level leave-one-out (Split A) alongside hole-level stratified holdout (Split B)that separates cross-condition generalisation from in-distribution performance
- A systematic ablation quantifying the individual contribution of peck selection, sensor/channel set, class-imbalance strategy, and input window size

- A practical feasibility case for near-real-time deployment: inference latency measured against actual peck duration, and training-cost comparisons across models

6.3 Limitations of the study

A few things limit how far these results can be pushed. The dataset comes from a single campaign and exactly one run, with no repeats for each of the nine cutting conditions. That's not much to lean on statistically, and it means there's no way to tell condition-to-condition variability apart from ordinary experimental noise. This lack of data is also a constraint in the CNN's performance since deep networks are typically far more data-hungry than tree-based ones. Everything was run on one machine, with one tool, on one material (AISI 316L). So the conclusions here hold for that specific combination and shouldn't be assumed to carry over to a different machine, tool, or material without checking first.

The CNN's epoch count for the full grid was fixed ahead of time from a single epoch-finder run, not tuned per configuration. A different epoch budget might have produced better CNN numbers than those reported; this is also true if a different random initial state was used.

None of the three models had their hyperparameters formally tuned; they were all fixed by hand rather than selected through a systematic search such as grid or random search with cross-validation. The comparison between models in this chapter is therefore a comparison between three reasonable, hand-picked configurations, not necessarily their best achievable ones; a properly tuned RF or XGBoost could close part of the gap to the CNN on Split B, or a tuned CNN could narrow the gap to the classical models on Split A.

And the flank wear values for the peck's label aren't direct measurements; they come from linear interpolation between V_B readings taken every ten holes. That leaves some uncertainty in the labels themselves, especially near the class boundaries.

6.4 Future research directions

A few directions are worth pursuing next. On the modelling side, there is room to explore deeper architectures, particularly LSTMs, either on their own or paired with the convolutional branches already built here. Self-supervised approaches are also worth a look: since labelled wear measurements are far scarcer than raw sensor recordings, pretraining an encoder on the unlabelled signal through a pretext task such as reconstruction or contrastive learning, and only then fine-tuning it for wear classification, could make better use of the data that is actually available. So is reframing the task as regression, predicting V_B directly instead of a wear class, as it may be more useful in an industrial application.

The denoising pipeline is a clear candidate for rework. As the ablation study showed, it currently yields only marginal gains over the raw signal rather than the significant improvement one would expect from stripping out non-wear-related noise, so its design is worth revisiting rather than taking as final.

The CNN's epoch budget is another area to revisit. In this work, a single epoch-finder probe set one fixed epoch count for every fold in the grid, a fairly coarse approach. Per-fold early stopping, monitored on a validation subset carved out of that fold's own training files, would let the epoch count adapt fold by fold instead of using one blanket number, and would likely narrow the gap between the CNN's best and worst folds seen in 5.4. This validation subset has to come from the training files, not the fold's held-out test file; using the test file for early-stopping decisions, rather than only for final evaluation, would leak test information into training and inflate the CNN's apparent performance.

On the engineering side, the pipeline itself has room to be optimised, on both the hardware and software fronts, which would go a long way toward cutting the CNN's training time and making it easier to deploy on more modest hardware.

The channel-set ablation only tested single sensor groups against the full set. It would be worth testing a handful of practical two- or three-sensor combinations too, to find a middle ground between the full multi-sensor setup and a single, cheaper sensor.

The dataset needs to be expanded with more experiments. With one file per cutting condition, there's no real way to say how much variation comes from the condition itself versus noise; repeating each combination would fix that.

And, further down the line, there's the step from research prototype to something that could actually run on a shop floor.

References

[1] S. Engin and Y. Altintas. “Mechanics and dynamics of general milling cutters. Part I: helical end mills”. In: *International Journal of Machine Tools and Manufacture* (2001).

[2] Roberto Munaro, Aldo Attanasio, and Antonio Del Prete. “Tool Wear Monitoring with Artificial Intelligence Methods: A Review”. In: *Journal of Manufacturing and Materials Processing* (2023).

[3] Lorenzo Colantonio et al. “A Systematic Literature Review of Cutting Tool Wear Monitoring in Turning by Using Artificial Intelligence Techniques”. In: *Machines* (2021).

[4] Xuebing Li et al. “Systematic review on tool breakage monitoring techniques in machining operations”. In: *International Journal of Machine Tools and Manufacture* (2022).

[5] S. Kurada and C. Bradley. “A review of machine vision sensors for tool condition monitoring”. In: *Computers in Industry* (1996).

[6] Chen Zhang and Jilin Zhang. “On-line tool wear measurement for ball-end milling cutter based on machine vision”. In: *Computers in Industry* (2013).

[7] L. Norberto López De Lacalle et al. “Using high pressure coolant in the drilling and turning of low machinability alloys”. In: *International Journal of Advanced Manufacturing Technology* (2000).

[8] David A. Stephenson and John S. Agapiou. *Metal Cutting Theory and Practice*. CRC Press, 2016.

[9] Qinglong An et al. “A data-driven model for milling tool remaining useful life prediction with convolutional and stacked LSTM network”. In: *Measurement* (2020).

[10] Hakan Wiklund. “Bayesian and Regression Approaches to On-Line Prediction of Residual Tool Life”. In: *Quality and Reliability Engineering International* (1998).

[11] Nitin Ambhore et al. “Tool Condition Monitoring System: A Review”. In: *4th International Conference on Materials Processing and Characterization*. 2015.

[12] A. F. V. Pedroso et al. “A Comprehensive Review on the Conventional and Non-Conventional Machining and Tool-Wear Mechanisms of INCONEL®”. In: *Metals* (2023).

[13] Vitalii Ivanov et al. “Machining Methods”. In: *Fundamentals of Manufacturing Engineering Using Digital Visualization*. Springer Nature Switzerland, 2025.

[14] Satish Chinchanikar and S. K. Choudhury. “Characteristic of Wear, Force and their Interrelationship: In-process Monitoring of Tool within Different Phases of the Tool Life”. In: *Procedia Materials Science* (2014).

[15] Dimla E. Dimla. “Sensor signals for tool-wear monitoring in metal cutting operations - a review of methods”. In: *International Journal of Machine Tools and Manufacture* (2000).

[16] Bernhard Sick. “On-line and indirect tool wear monitoring in turning with artificial neural networks: a review of more than a decade of research”. In: *Mechanical Systems and Signal Processing* (2002).

[17] MAS Tools. *Tool wear in machining: understanding mechanisms, optimising tool life*. 2026. URL: https://www.mas-tools.de/en/solutions/process-optimization/tool-wear.

[18] Bikash Chandra Behera et al. “The underlying mechanisms of coolant contribution in the machining process”. In: *Machining and Tribology: Processes, Surfaces, Coolants, and Modeling*. Elsevier, 2022.

[19] Danil Yu. Pimenov et al. “Review of improvement of machinability and surface integrity in machining on aluminum alloys”. In: *The International Journal of Advanced Manufacturing Technology* (2023).

[20] P. J. Arrazola et al. “Machinability of titanium alloys (Ti6Al4V and Ti555.3)”. In: *Journal of Materials Processing Technology* (2009).

[21] International Organization for Standardization. *Tool life testing in milling Part 1: Face milling*. 2022. URL: https://www.iso.org/standard/16091.html.

[22] Kistler Group. *Kistler - Measurement Technology for the World*. 2026. URL: https://www.kistler.com.

[23] LEM. *LEM - Current and Voltage Transducers*. 2026. URL: https://www.lem.com.

[24] Frederick W. Taylor. “On the Art of Cutting Metals”. In: *Transactions of the ASME* (1907).

[25] N. E. Woldman and R. C. Gibbons. *Machinability and Machining of Metals*. McGraw-Hill, 1951.

[26] H. Attia et al. “Physics based models for characterization of machining performance - A critical review”. In: *CIRP Journal of Manufacturing Science and Technology* (2024).

[27] Leo Breiman. “Random Forests”. In: *Machine Learning* (2001).

[28] Dazhong Wu et al. “A Comparative Study on Machine Learning Algorithms for Smart Manufacturing: Tool Wear Prediction Using Random Forests”. In: *Journal of Manufacturing Science and Engineering* (2017).

[29] Dazhong Wu et al. “Cloud-based machine learning for predictive analytics: Tool wear prediction in milling”. In: *2016 IEEE International Conference on Big Data*. 2016.

[30] Corinna Cortes and Vladimir Vapnik. “Support-vector networks”. In: *Machine Learning* (1995).

[31] Achyuth Kothuru, Sai Prasad Nooka, and Rui Liu. “Application of audible sound signals for tool wear monitoring using machine learning techniques in end milling”. In: *International Journal of Advanced Manufacturing Technology* (2018).

[32] Milla Caroline Gomes et al. “Tool wear monitoring in micromilling using Support Vector Machine with vibration and sound sensor”. In: *Precision Engineering* (2021).

[33] L. R. Rabiner. “A tutorial on hidden Markov models and selected applications in speech recognition”. In: *Proceedings of the IEEE* (1989).

[34] Ming Dong and David He. “Hidden semi-Markov model-based methodology for multi-sensor equipment health diagnosis and prognosis”. In: *European Journal of Operational Research* (2007).

[35] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* (1998).

[36] Achyuth Kothuru, Sai Prasad Nooka, and Rui Liu. “Application of deep visualization in CNN-based tool condition monitoring for end milling”. In: *Procedia Manufacturing* (2019).

[37] Clayton Cooper et al. “Convolutional neural network-based tool condition monitoring in vertical milling operations using acoustic signals”. In: *Procedia Manufacturing* (2020).

[38] Jeffrey L. Elman. “Finding Structure in Time”. In: *Cognitive Science* (1990).

[39] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* (1997).

[40] Weili Cai et al. “A hybrid information model based on long short-term memory network for tool condition monitoring”. In: *Journal of Intelligent Manufacturing* (2020).

[41] Milind Shah et al. “Tool wear prediction in face milling of stainless steel using singular generative adversarial network and LSTM deep learning models”. In: *The International Journal of Advanced Manufacturing Technology* (2022).

[42] Zhaopeng He, Tielin Shi, and Jianping Xuan. “Milling tool wear prediction using multi-sensor feature fusion based on stacked sparse autoencoders”. In: *Measurement* (2022).

[43] Lei Wang et al. “A gated graph convolutional network with multi-sensor signals for remaining useful life prediction”. In: *Knowledge-Based Systems* (2022).

[44] Jawad Mahmood, Ming Luo, and Mudassar Rehman. “An accurate detection of tool wear type in drilling process by applying PCA and one-hot encoding to SSA-BLSTM model”. In: *The International Journal of Advanced Manufacturing Technology* (2022).

[45] Soumya Ashwath et al. “Deep Hybrid CNN-LSTM Framework for Accurate Tool Wear Prediction in Machining Inconel 718”. In: *Proceedings of the 3rd International Conference on IoT, Communication and Automation Technology (ICICAT)*. 2025.

[46] Amuthakkannan Rajakannu et al. “Intelligent Tool Wear Classification of a CNC Drill Bit Using Feature Fusion and a Family of Lazy Classifiers”. In: *SSRG International Journal of Mechanical Engineering* (2025).

[47] Amuthakkannan Rajakannu et al. “ML-Based Advanced Electrical Signal Decomposition using the Hilbert-Huang Transform and CatBoost Classification for Acoustic Signal-Driven Smart Tool Wear Detection”. In: *SSRG International Journal of Electrical and Electronics Engineering* (2026).

[48] Roberto Munaro et al. “A New Architecture Paradigm for Tool Wear Prediction during AISI 9840 Drilling Operation”. In: *Procedia Computer Science*. 2024.

[49] Danil Yu. Pimenov et al. “Artificial intelligence systems for tool condition monitoring in machining: analysis and critical review”. In: *Journal of Intelligent Manufacturing* (2023).

[50] Haoyuan Zhang et al. “A Review of Physics-Based, Data-Driven, and Hybrid Models for Tool Wear Monitoring”. In: *Machines* (2024).

[51] Haas Automation. *VF-2 Vertical Machining Center*. 2026. URL: https://www.haascnc.com/machines/vertical-mills/vf-series/models/small/vf-2.html.

[52] PALBIT. *General Catalogue 2024*. 2026. URL: https://online.flippingbook.com/view/834871936/.

[53] YG-1. *Solid Carbide Drill DL105060 - Product Catalogue*. 2026. URL: https://www.yg1.solutions/toolselection/DRILLING/detail.asp.

[54] C. Mãranhão and J. Paulo Davim. “Finite element modelling of machining of AISI 316 steel: Numerical simulation and experimental validation”. In: *Simulation Modelling Practice and Theory* (2010).

[55] Advantech. *USB-4704 Data Acquisition Module*. 2026. URL: https://www.advantech.com/pt-br/products/1-2mlkno/usb-4704/mod_4d0800cc-f6fd-402a-9782-24cd0ffdaf42.

[56] A. Z. Sultan, Safian Sharif, and Denni Kurniawan. “Effect of Machining Parameters on Tool Wear and Hole Quality of AISI 316L Stainless Steel in Conventional Drilling”. In: *Procedia Manufacturing* (2015).

[57] Dino-Lite. *Dino-Lite Digital Microscopes*. 2026. URL: https://www.dino-lite.com.

Appendix A

G-code used in the experiments

Three G-code programs were used to acquire the drilling dataset. TEST BASE NOISE.nc records baseline sensor noise (idle spindle and air-cutting, no workpiece contact) for the denoising pipeline. MAIN.nc stores the hole grid coordinates and calls the drilling subprogram for each point.SUB.nc performs the drilling cycle for each hole, controls spindle speed by hole count, and pauses every 10 holes for wear inspection. Both MAIN/SUB programs were reused with minor modifications for the carbide tool runs.

Listing A.1: TEST BASE NOISE.nc

```
%
O06660 (Checks program)

G90 G17 G21
G53 G0 Z0.
G54
T12 M06 (Chose tool)
G43 H12 Z45.
G95 (F mm/rev)

(DAQ and sensor base noise)
M59 P90 (DAQ ON)
G04 P10000 (10 sec)
M69 P90 (DAQ OFF)

G04 P2000

( Sweeping test sem coolant )
G0 X0. Y0.
G0 Z14.
S531 M3  (20)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
S797 M3 (40)
```

```
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
S1062 M3 (20)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
S1328 M3 (80)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
S1593 M3 (100)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)

M05

G04 P2000

M08 (Ligar o internal colant)

( Sweeping test com coolant )
G0 X0. Y0.
G0 Z10.
S531 M3  (20)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
S797 M3 (40)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
S1062 M3 (20)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
S1328 M3 (80)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)
```

```
S1593 M3 (100)
G04 P3000  (Ramp up time )
M59 P90 (DAQ ON)
G04 P10000  (Delay 10 sec )
M69 P90 (DAQ OFF)

M05

G04 P2000

M08









(--- AIR CUTTING SECTION 20 ---)
#601 = 797 (Speed RPM, start with the one you want for the air cutting)
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.06
G0 Z20.
M69 P90 (DAQ OFF)

G04 P2000

(--- AIR CUTTING SECTION 60 ---)
#601 = 1062
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.06
G0 Z20.
M69 P90 (DAQ OFF)

G04 P2000

(--- AIR CUTTING SECTION 80 ---)
#601 = 1328
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
```

```
M59 P90 (DAQ ON)
G1 Z14. F0.06
G0 Z20.
M69 P90 (DAQ OFF)



M05
G04 P2000
M08

(--- AIR CUTTING SECTION 20 ---)
#601 = 797 (Speed RPM, start with the one you want for the air cutting)
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.08
G0 Z20.
M69 P90 (DAQ OFF)

G04 P2000

(--- AIR CUTTING SECTION 60 ---)
#601 = 1062
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.08
G0 Z20.
M69 P90 (DAQ OFF)

G04 P2000

(--- AIR CUTTING SECTION 80 ---)
#601 = 1328
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.08
G0 Z20.
M69 P90 (DAQ OFF)


```

```
M05
G04 P2000
M08

(--- AIR CUTTING SECTION 20 ---)
#601 = 797 (Speed RPM, start with the one you want for the air cutting)
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.1
G0 Z20.
M69 P90 (DAQ OFF)

G04 P2000

(--- AIR CUTTING SECTION 60 ---)
#601 = 1062
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.1
G0 Z20.
M69 P90 (DAQ OFF)

G04 P2000

(--- AIR CUTTING SECTION 80 ---)
#601 = 1328
G0 Z20.
G0 X0. Y0.
S#601 M3 (Turn spindle on 40)
G04 P2000 (Ramp up time)
M59 P90 (DAQ ON)
G1 Z14. F0.1
G0 Z20.
M69 P90 (DAQ OFF)


G04 P2000


(End drilling)
M5 (stop spindle)
M9 (Stop all cooling )
G53 G0 Z0.
```

```
G53 G0 X0. Y0.
M30 (end program)
%
```

Listing A.2: MAIN.nc

```
%
O01010 (GRID DRILLING WITH WEAR CHECK EVERY 10 HOLES)

G90 G17 G21
G53 G0 Z0.
G54
T12 M06 (Chose tool)
G43 H12 Z45.
G95 (F mm/rev)


M08 (Ligar o internal colant)


( Variables )
#600 = 0 (Number of holes)
#601 = 1328 (Speed RPM, start with the one you want for the air cutting)
#602 = 0 (Speed of the last hole)

S#601 M3
(Calls subprogram using HOLE COORDINATES)


G65 P7707 X-10.95 Y-69.35
G65 P7707 X-3.65 Y-69.35
G65 P7707 X3.65 Y-69.35
G65 P7707 X10.95 Y-69.35
G65 P7707 X-32.85 Y-62.05
G65 P7707 X-25.55 Y-62.05
G65 P7707 X-18.25 Y-62.05
G65 P7707 X-10.95 Y-62.05
G65 P7707 X-3.65 Y-62.05
G65 P7707 X3.65 Y-62.05

G65 P7707 X10.95 Y-62.05
G65 P7707 X18.25 Y-62.05
G65 P7707 X25.55 Y-62.05
G65 P7707 X32.85 Y-62.05
G65 P7707 X-40.15 Y-54.75
G65 P7707 X-32.85 Y-54.75
G65 P7707 X-25.55 Y-54.75
G65 P7707 X-18.25 Y-54.75
G65 P7707 X-10.95 Y-54.75
G65 P7707 X-3.65 Y-54.75
```

```
45
46 G65 P7707 X3.65 Y-54.75
47 G65 P7707 X10.95 Y-54.75
48 G65 P7707 X18.25 Y-54.75
49 G65 P7707 X25.55 Y-54.75
50 G65 P7707 X32.85 Y-54.75
51 G65 P7707 X40.15 Y-54.75
52 G65 P7707 X-47.45 Y-47.45
53 G65 P7707 X-40.15 Y-47.45
54 G65 P7707 X-32.85 Y-47.45
55 G65 P7707 X-25.55 Y-47.45
56
57 G65 P7707 X-18.25 Y-47.45
58 G65 P7707 X-10.95 Y-47.45
59 G65 P7707 X-3.65 Y-47.45
60 G65 P7707 X3.65 Y-47.45
61 G65 P7707 X10.95 Y-47.45
62 G65 P7707 X18.25 Y-47.45
63 G65 P7707 X25.55 Y-47.45
64 G65 P7707 X32.85 Y-47.45
65 G65 P7707 X40.15 Y-47.45
66 G65 P7707 X47.45 Y-47.45
67
68 G65 P7707 X-54.75 Y-40.15
69 G65 P7707 X-47.45 Y-40.15
70 G65 P7707 X-40.15 Y-40.15
71 G65 P7707 X-32.85 Y-40.15
72 G65 P7707 X-25.55 Y-40.15
73 G65 P7707 X-18.25 Y-40.15
74 G65 P7707 X-10.95 Y-40.15
75 G65 P7707 X-3.65 Y-40.15
76 G65 P7707 X3.65 Y-40.15
77 G65 P7707 X10.95 Y-40.15
78
79 G65 P7707 X18.25 Y-40.15
80 G65 P7707 X25.55 Y-40.15
81 G65 P7707 X32.85 Y-40.15
82 G65 P7707 X40.15 Y-40.15
83 G65 P7707 X47.45 Y-40.15
84 G65 P7707 X54.75 Y-40.15
85 G65 P7707 X-62.05 Y-32.85
86 G65 P7707 X-54.75 Y-32.85
87 G65 P7707 X-47.45 Y-32.85
88 G65 P7707 X-40.15 Y-32.85
89
90 G65 P7707 X-32.85 Y-32.85
91 G65 P7707 X-25.55 Y-32.85
92 G65 P7707 X-18.25 Y-32.85
93 G65 P7707 X-10.95 Y-32.85
```

```
G65 P7707 X-3.65 Y-32.85
G65 P7707 X3.65 Y-32.85
G65 P7707 X10.95 Y-32.85
G65 P7707 X18.25 Y-32.85
G65 P7707 X25.55 Y-32.85
G65 P7707 X32.85 Y-32.85

G65 P7707 X40.15 Y-32.85
G65 P7707 X47.45 Y-32.85
G65 P7707 X54.75 Y-32.85
G65 P7707 X62.05 Y-32.85
G65 P7707 X-62.05 Y-25.55
G65 P7707 X-54.75 Y-25.55
G65 P7707 X-47.45 Y-25.55
G65 P7707 X-40.15 Y-25.55
G65 P7707 X-32.85 Y-25.55
G65 P7707 X-25.55 Y-25.55

G65 P7707 X-18.25 Y-25.55
G65 P7707 X-10.95 Y-25.55
G65 P7707 X-3.65 Y-25.55
G65 P7707 X3.65 Y-25.55
G65 P7707 X10.95 Y-25.55
G65 P7707 X18.25 Y-25.55
G65 P7707 X25.55 Y-25.55
G65 P7707 X32.85 Y-25.55
G65 P7707 X40.15 Y-25.55
G65 P7707 X47.45 Y-25.55

G65 P7707 X54.75 Y-25.55
G65 P7707 X62.05 Y-25.55
G65 P7707 X-62.05 Y-18.25
G65 P7707 X-54.75 Y-18.25
G65 P7707 X-47.45 Y-18.25
G65 P7707 X-40.15 Y-18.25
G65 P7707 X-32.85 Y-18.25
G65 P7707 X-25.55 Y-18.25
G65 P7707 X-18.25 Y-18.25
G65 P7707 X-10.95 Y-18.25

G65 P7707 X-3.65 Y-18.25
G65 P7707 X3.65 Y-18.25
G65 P7707 X10.95 Y-18.25
G65 P7707 X18.25 Y-18.25
G65 P7707 X25.55 Y-18.25
G65 P7707 X32.85 Y-18.25
G65 P7707 X40.15 Y-18.25
G65 P7707 X47.45 Y-18.25
G65 P7707 X54.75 Y-18.25
```

```
143 G65 P7707 X62.05 Y-18.25
144
145 G65 P7707 X-69.35 Y-10.95
146 G65 P7707 X-62.05 Y-10.95
147 G65 P7707 X-54.75 Y-10.95
148 G65 P7707 X-47.45 Y-10.95
149 G65 P7707 X-40.15 Y-10.95
150 G65 P7707 X-32.85 Y-10.95
151 G65 P7707 X-25.55 Y-10.95
152 G65 P7707 X-18.25 Y-10.95
153 G65 P7707 X-10.95 Y-10.95
154 G65 P7707 X-3.65 Y-10.95
155
156 G65 P7707 X3.65 Y-10.95
157 G65 P7707 X10.95 Y-10.95
158 G65 P7707 X18.25 Y-10.95
159 G65 P7707 X25.55 Y-10.95
160 G65 P7707 X32.85 Y-10.95
161 G65 P7707 X40.15 Y-10.95
162 G65 P7707 X47.45 Y-10.95
163 G65 P7707 X54.75 Y-10.95
164 G65 P7707 X62.05 Y-10.95
165 G65 P7707 X69.35 Y-10.95
166
167 G65 P7707 X-69.35 Y-3.65
168 G65 P7707 X-62.05 Y-3.65
169 G65 P7707 X-54.75 Y-3.65
170 G65 P7707 X-47.45 Y-3.65
171 G65 P7707 X-40.15 Y-3.65
172 G65 P7707 X-32.85 Y-3.65
173 G65 P7707 X-25.55 Y-3.65
174 G65 P7707 X-18.25 Y-3.65
175 G65 P7707 X-10.95 Y-3.65
176 G65 P7707 X-3.65 Y-3.65
177
178 G65 P7707 X3.65 Y-3.65
179 G65 P7707 X10.95 Y-3.65
180 G65 P7707 X18.25 Y-3.65
181 G65 P7707 X25.55 Y-3.65
182 G65 P7707 X32.85 Y-3.65
183 G65 P7707 X40.15 Y-3.65
184 G65 P7707 X47.45 Y-3.65
185 G65 P7707 X54.75 Y-3.65
186 G65 P7707 X62.05 Y-3.65
187 G65 P7707 X69.35 Y-3.65
188
189 G65 P7707 X-69.35 Y3.65
190 G65 P7707 X-62.05 Y3.65
191 G65 P7707 X-54.75 Y3.65
```

```
G65 P7707 X-47.45 Y3.65
G65 P7707 X-40.15 Y3.65
G65 P7707 X-32.85 Y3.65
G65 P7707 X-25.55 Y3.65
G65 P7707 X-18.25 Y3.65
G65 P7707 X-10.95 Y3.65
G65 P7707 X-3.65 Y3.65
G65 P7707 X3.65 Y3.65
G65 P7707 X10.95 Y3.65
G65 P7707 X18.25 Y3.65
G65 P7707 X25.55 Y3.65
G65 P7707 X32.85 Y3.65
G65 P7707 X40.15 Y3.65
G65 P7707 X47.45 Y3.65
G65 P7707 X54.75 Y3.65
G65 P7707 X62.05 Y3.65
G65 P7707 X69.35 Y3.65

G65 P7707 X-69.35 Y10.95
G65 P7707 X-62.05 Y10.95
G65 P7707 X-54.75 Y10.95
G65 P7707 X-47.45 Y10.95
G65 P7707 X-40.15 Y10.95
G65 P7707 X-32.85 Y10.95
G65 P7707 X-25.55 Y10.95
G65 P7707 X-18.25 Y10.95
G65 P7707 X-10.95 Y10.95
G65 P7707 X-3.65 Y10.95
G65 P7707 X3.65 Y10.95
G65 P7707 X10.95 Y10.95
G65 P7707 X18.25 Y10.95
G65 P7707 X25.55 Y10.95
G65 P7707 X32.85 Y10.95
G65 P7707 X40.15 Y10.95
G65 P7707 X47.45 Y10.95
G65 P7707 X54.75 Y10.95
G65 P7707 X62.05 Y10.95
G65 P7707 X69.35 Y10.95

G65 P7707 X-62.05 Y18.25
G65 P7707 X-54.75 Y18.25
G65 P7707 X-47.45 Y18.25
G65 P7707 X-40.15 Y18.25
G65 P7707 X-32.85 Y18.25
G65 P7707 X-25.55 Y18.25
G65 P7707 X-18.25 Y18.25
G65 P7707 X-10.95 Y18.25
G65 P7707 X-3.65 Y18.25
G65 P7707 X3.65 Y18.25
```

```
241 G65 P7707 X10.95 Y18.25
242 G65 P7707 X18.25 Y18.25
243 G65 P7707 X25.55 Y18.25
244 G65 P7707 X32.85 Y18.25
245 G65 P7707 X40.15 Y18.25
246 G65 P7707 X47.45 Y18.25
247 G65 P7707 X54.75 Y18.25
248 G65 P7707 X62.05 Y18.25
249
250 G65 P7707 X-62.05 Y25.55
251 G65 P7707 X-54.75 Y25.55
252 G65 P7707 X-47.45 Y25.55
253 G65 P7707 X-40.15 Y25.55
254 G65 P7707 X-32.85 Y25.55
255 G65 P7707 X-25.55 Y25.55
256 G65 P7707 X-18.25 Y25.55
257 G65 P7707 X-10.95 Y25.55
258 G65 P7707 X-3.65 Y25.55
259 G65 P7707 X3.65 Y25.55
260 G65 P7707 X10.95 Y25.55
261 G65 P7707 X18.25 Y25.55
262 G65 P7707 X25.55 Y25.55
263 G65 P7707 X32.85 Y25.55
264 G65 P7707 X40.15 Y25.55
265 G65 P7707 X47.45 Y25.55
266 G65 P7707 X54.75 Y25.55
267 G65 P7707 X62.05 Y25.55
268
269 G65 P7707 X-62.05 Y32.85
270 G65 P7707 X-54.75 Y32.85
271 G65 P7707 X-47.45 Y32.85
272 G65 P7707 X-40.15 Y32.85
273 G65 P7707 X-32.85 Y32.85
274 G65 P7707 X-25.55 Y32.85
275 G65 P7707 X-18.25 Y32.85
276 G65 P7707 X-10.95 Y32.85
277 G65 P7707 X-3.65 Y32.85
278 G65 P7707 X3.65 Y32.85
279 G65 P7707 X10.95 Y32.85
280 G65 P7707 X18.25 Y32.85
281 G65 P7707 X25.55 Y32.85
282 G65 P7707 X32.85 Y32.85
283 G65 P7707 X40.15 Y32.85
284 G65 P7707 X47.45 Y32.85
285 G65 P7707 X54.75 Y32.85
286 G65 P7707 X62.05 Y32.85
287
288 G65 P7707 X-54.75 Y40.15
289 G65 P7707 X-47.45 Y40.15
```

```
G65 P7707 X-40.15 Y40.15
G65 P7707 X-32.85 Y40.15
G65 P7707 X-25.55 Y40.15
G65 P7707 X-18.25 Y40.15
G65 P7707 X-10.95 Y40.15
G65 P7707 X-3.65 Y40.15
G65 P7707 X3.65 Y40.15
G65 P7707 X10.95 Y40.15
G65 P7707 X18.25 Y40.15
G65 P7707 X25.55 Y40.15
G65 P7707 X32.85 Y40.15
G65 P7707 X40.15 Y40.15
G65 P7707 X47.45 Y40.15
G65 P7707 X54.75 Y40.15

G65 P7707 X-47.45 Y47.45
G65 P7707 X-40.15 Y47.45
G65 P7707 X-32.85 Y47.45
G65 P7707 X-25.55 Y47.45
G65 P7707 X-18.25 Y47.45
G65 P7707 X-10.95 Y47.45
G65 P7707 X-3.65 Y47.45
G65 P7707 X3.65 Y47.45
G65 P7707 X10.95 Y47.45
G65 P7707 X18.25 Y47.45
G65 P7707 X25.55 Y47.45
G65 P7707 X32.85 Y47.45
G65 P7707 X40.15 Y47.45
G65 P7707 X47.45 Y47.45

G65 P7707 X-40.15 Y54.75
G65 P7707 X-32.85 Y54.75
G65 P7707 X-25.55 Y54.75
G65 P7707 X-18.25 Y54.75
G65 P7707 X-10.95 Y54.75
G65 P7707 X-3.65 Y54.75
G65 P7707 X3.65 Y54.75
G65 P7707 X10.95 Y54.75
G65 P7707 X18.25 Y54.75
G65 P7707 X25.55 Y54.75
G65 P7707 X32.85 Y54.75
G65 P7707 X40.15 Y54.75

G65 P7707 X-32.85 Y62.05
G65 P7707 X-25.55 Y62.05
G65 P7707 X-18.25 Y62.05
G65 P7707 X-10.95 Y62.05
G65 P7707 X-3.65 Y62.05
G65 P7707 X3.65 Y62.05
```

```
G65 P7707 X10.95 Y62.05
G65 P7707 X18.25 Y62.05
G65 P7707 X25.55 Y62.05
G65 P7707 X32.85 Y62.05

G65 P7707 X-10.95 Y69.35
G65 P7707 X-3.65 Y69.35
G65 P7707 X3.65 Y69.35
G65 P7707 X10.95 Y69.35

(End drilling)
M5 (stop spindle)
M9 (Stop all cooling )
G53 G0 Z0.
G53 G0 X0. Y0.
M30 (end program)
%
```

Listing A.3: SUB.nc

```
%
O08210 (DRILL ONE HOLE AT X/Y)

( X = #24, Y = #25 )
G95 ( Re-enforce mm/rev)
G0 X#24 Y#25
G0 Z5. (Ensure the position 5 mm above the hole center for inicial hole and after
    wear check)

#600 = #600 + 1      (Increment hole counter)
#602 = #601          (Store previous spindle speed)

(--- SPINDLE SPEED CONTROL ---)
IF [#600 LE 10] GOTO 10
IF [#600 LE 20] GOTO 20
IF [#600 LE 30] GOTO 30

(variables)
N10 #601 = 1328 (1-10 Holes RPM)
GOTO 40

N20 #601 = 1328 (11-20 Holes RPM)
GOTO 40

N30 #601 = 1328 (21-30 Holes RPM)

N40
IF [#601 EQ #602] GOTO 50
S#601 M3
G04 P2500  (Ramp up time )
```

```
N50


M59 P90          (DAQ ON)

G98 G83 X#24 Y#25 Z-24.0 R1.0 Q6.0 F0.1
G80
G0 Z5.
M69 P90          (DAQ OFF)

G04 P500


(--- WEAR CHECK EVERY 10 HOLES ---)
IF [[#600 MOD 10] NE 0] GOTO 99


IF [#600 EQ 30] #600 = 0
M05
G0 Z80.0 ( Send Tool to a location for easy access)
M00 (CHECK DRILL WEAR)
M08
G0 Z5.0 ( Send Tool to the previouse location )
S#601 M3
G04 P1500 (Delay 1500ms )

N99 M99
%
```