Title

Industrial Data Reduction, Aggregation and Machine Learning-Based Soft Sensing for Etching and Slider Production Tools

Author

Suherman, Julius Owen

Publication Date

2025

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Industrial Data Reduction, Aggregation and Machine Learning-Based Soft Sensing for Etching and Slider Production Tools

A thesis submitted in partial satisfaction
of the requirements for the degree Master of Science
in Chemical Engineering

by

Julius Owen Suherman

2025

ABSTRACT OF THE THESIS

Industrial Data Reduction, Aggregation and Machine Learning-Based Soft Sensing for Etching and Slider Production Tools

by

Julius Owen Suherman
Master of Science in Chemical Engineering
University of California, Los Angeles, 2025
Professor Panagiotis D. Christofides, Chair

Smart Manufacturing (SM), which is short for "Smart (Predictive, Preventive, Proactive) zero incident, zero emissions Manufacturing," describes manufacturing's digital transformation in which factories, supply chains and ecosystems are integrated, interoperable, and interconnected. Smart Manufacturing is rooted in AI, Machine Learned (ML), and Data Synchronized (DS) modeling to tap into invaluable operating data. By making data actionable at larger scales, SM opens new ways to increase productivity, precision, and process performance. Smart Manufacturing applied to front-end wafer manufacturing in the semiconductor industry offers significant opportunity to increase production throughput and ensure precision by increasing staff and operational productivity. Front-end wafer manufacturing involves multi tool operations for complex material processing that requires a high degree of precision and extensive product qualification. There is a high degree of commonality with semiconductor manufacturing tools, for example etching, that are well instrumented. Companies are already collecting large amounts of operational data from these tools

that can be aggregated and leveraged for virtual metrology and other control, diagnostic, and management solutions. AI/ML/DS modeling involves monitoring the state of an operation in real-time to continuously learn and improve on human centered, automated, and autonomous actions. This operational data are embedded in invaluable machine, process, product, and material behaviors as interaction complexities, linearities/non-linearities, and dimensional effects. Because of machine commonalities, data can be selected to draw out operational value across machines. Today's data science offers considerable capability for qualifying, assessing alignment and contribution, aggregating, and engineering data for more robust modeling. We refer to this as a Data-first strategy to process, engineer and model with AI-Ready data. In this paper, we address AI-Ready data for a virtual metrology solution focused on etching measurement PASS/FAIL classification and milling depth prediction regression tasks using operational data from production machine tools. If the quality of the product can be predicted, the productivity of the metrology process can be increased, which in turn increases the productivity of the overall operation. In a previous paper, we considered how to aggregate data from different etch tools in the same processes at different factories within Seagate Technology and proposed a method for data aggregation and demonstrated its value [1]. The present paper considers how to process and engineer datasets from two different etch tool processes: wafer and slider production. The data processing approaches when used systematically with appropriate ML algorithms demonstrate the potential for reducing metrological interventions in semiconductor manufacturing. Advanced machine learning techniques are used to tackle the modeling challenges of a low failure rate and limited operational variability. XGBoost, a gradient descent-based tree algorithm, outperforms the commonly used Feedforward Neural Networks (FNN) in terms of training speed and resource utilization for binary-classifications. Principal Component Analysis (PCA) effectively reduces the dimensionality of the data and overfitting, while retaining vital variances and significantly reducing noise. Data aggregation with separated scaling harmonizes inputs from diverse manufacturing tools and significantly improves the efficacy and versatility of combining multiple datasets to improve model performance. A live updating

transfer learning approach, that periodically updates the FNN models in real-time using Stochastic Gradient Descent (SGD) with individual data points, addresses process drift, and markedly improves predictive accuracy. For the slider production tools, data augmentation with linear Mixup, overcomes a short recording period, enriches the training dataset, and significantly reduces error metrics.

The thesis of Julius Owen Suherman is approved.

Dante A. Simonetti

Philippe Sautet

Panagiotis D. Christofides, Committee Chair

University of California, Los Angeles

2025

Contents

List of Figures

List of Tables

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Panagiotis D. Christofides, for his guidance and support during the course of my research.

I would like to thank Professors Philippe Sautet and Dante Simonetti for reviewing my thesis and contributing to my Master's thesis committee.

I would like to gratefully acknowledge financial support from the U.S. Department of Energy, through the Office of Energy Efficiency and Renewable Energy (EERE), under the Advanced Manufacturing Office Award Number DE-EE0007613.

I would like to thank Henrik Wang and Feiyang Ou for their guidance, comments, and support toward this research.

I would like to thank Seagate Technology and CESMII for providing the industrial data utilized in this project.

Lastly, I would like to thank my father, Sukiman Suherman, mother, Grace Munando, and brother, Basilio Suherman, along with all my friends who have given me tremendous support.

Chapter 1

Introduction

As the transition to the digital era accelerates and the Internet of Things (IoT) grows, the demand for microelectronic devices is skyrocketing. The Semiconductor Industry Association (SIA) announced that global semiconductor industry sales totaled $574.1 billion in 2022, the highest ever annual total and an increase of 3.3% compared to the 2021 total of $555.9 billion. The industry shipped a record 1.15 trillion semiconductor units in 2021, as chip companies ramped up production to address high demand amid the global chip shortage [2]. These are devices equipped with integrated circuits such as central processing units (CPUs), graphics processing units (GPUs), hard drives, and solid state drives [3]. Demand for these devices has long driven the need for higher transistor densities and narrower gate widths to improve computing performance and reduce power consumption rates [4]. Today, the increase in demand has led to recurring shortages [5], negatively affecting global economies in various sectors, for example, the automotive industry [6]. There is a substantial incentive in semiconductor manufacturing to increase production volume, increase product quality and precision, and increase productivity without only increasing operational equipment and personnel. Digital transformation and leveraging Industry 4.0 for Smart Manufacturing are pivotal with how to take advantage of operational data, AI/ML, IoT, information technology, and interconnectedness at greater scales to open new avenues of increased productivity, perfor-

mance and precision [7]. Smart manufacturing is, by definition, about using real-time data and modeling at scale. It is the intersection of plant-wide agility and optimization, sustainable production and resilient demand-driven supply chains made interoperable through interconnectedness with trust and meaningfully shared data and tools [8].

Smart Manufacturing at scale depends on operating sensor data collected, contextualized, and aggregated from sensors on factory floor operations. Specifically, sensors are installed on each machine for each process operation, and they process and/or collect data in real time. These data are used to manage, control, and optimize the performance of each individual machine tool and process operation. They are useful for multiple management objectives such as monitoring, diagnosis, operational health, preventive maintenance, faster changeovers and quality assurance [9]. These data are critical to automation and autonomous operations. When used across line and factory operations, the data are used to address interoperability, agility, and higher-level key performance indicators in combination. Line operation and factory interoperability extend into supply chain interoperability. Cross company data that provide visibility into supply chain material and product flows and product demands and capacities are needed for resilience. In addition to the benefits of applying data and models to improve physical operations at all levels, these same data are also valuable assets that can be aggregated to build richer data sets for model building [1]. The data can be categorized, discovered and used to validate models and when synchronized with models, the resulting digital twin systems can be used for real-time preventive, proactive, predictive control, management and optimization [10].

In this paper we focus on data as valuable assets and demonstrate how data from multiple similar machines can be aggregated into richer datasets for more robust machine learning (ML) model building for all machines and/or how multiple machines can be optimized together. We focus on method and demonstration of virtual metrology (a.k.a., soft sensing) as one important application in semiconductor manufacturing. Unlike direct measurement methods, such as an offline quartz microbalance to analyze layer thickness manufacturing [11], ML virtual metrology

predicts product quality measurements or condition from operating sensor data by using these data to train models that can, within the range of operating conditions experienced, associate operating conditions to quality measurement [12]. The advantage of soft sensing lies in its ability to overcome the drawbacks of direct sensing, which can be expensive, time-consuming, untimely and/or labor-intensive [13]. There can be situations in which the measurement costs more than the product. There are also situations in which technical measurement complexity, product conditions, and or measurement objectives make physically impossible to do a direct measurement, e.g., key performance indicators (KPIs).

In contrast, virtual metrology uses data generated during production processes. These are data that embed operational effects for the range of operations experienced. The data, collected from widely available sensors are relatively inexpensive. The engineering of the data and the modeling process, if done systematically and carefully, has been shown to be highly effective in measuring physical parameters that are difficult to directly assess [14]. Benefits from these models can accrue quickly. However, these ML models are limited to operating range experienced. It is therefore crucial to pay close attention to the operating conditions within which the reliability of the model is high. Modern manufacturing processes have become increasingly complex, generating vast amounts of data from dozens or even hundreds of sensors [15]. There can be data volumes and operational complexities that exceed human assimilation. Machine learning and deep learning techniques have emerged as promising methods. They perform well in capturing complex non-linear correlations between inputs and outputs, and it has been demonstrated that neural networks of sufficient scale can approximate any non-linear functions [16].

Several types of neural networks have been successfully implemented in production settings: for example, Convolutional Neural Networks (CNNs) [17] and Recurrent Neural Networks (RNNs) [18]. Transformer networks have been used by companies including Seagate Technology for fault detection in etch tools [19]. Although recent advancements in ML have prompted extensive research into the application of soft sensing across various industries, challenges such as data

insufficiency, sensor noise, and the presence of redundant sensors remain significant problems. Furthermore, the issue of process drift over time complicates the effectiveness and reliability of models built on historical data.

In this paper, ML virtual metrology is tailored for both plasma etching and slider production. Specifically, datasets from multiple plasma etching and slider production tools were consistently collected and contextualized by applying CESMII's Data Information Model structure, called a Profile [8]. The Profile ensures that the measurements from each of the machines could be concatenated for further cross-machine analyses and uses of the data. Importantly, the paper recognizes that these AI/ML/DS models depend on, in fact thrive on the "right" data when there is enough that is engineered appropriately to build, train, test, and validate the AI/ML models. A "Data-First" strategy emphasizes the importance of data available in the amounts, forms, scale, and access necessary to achieve benefits in productivity, jobs, market share, sustainability, and growth. A Data-First strategy emphasizes engineering AI-Ready data that is consistently, collected, ingested, contextualized, cleaned, normalized, protected, aggregated, assessed, and selected to draw out the operational value that has been refined from years of experience even when the physics may not be fully understood. Data-First also addresses the primacy of data access in implementing affordable AI/ML solutions including how to obtain, categorize, scale, and use AI-Ready data. This includes how to validate, learn, unlearn, and adjust models when using and reusing this invaluable data.

This paper explores the impact of data modification and manipulation techniques, including dimension reduction, data aggregation, and data augmentation, on model performance to address these challenges. Principal Component Analysis (PCA) is employed as a dimension reduction technique to eliminate redundant features and reduce noise. Building upon a prior work [1], this study introduces a separate scaling method designed to normalize input datasets more effectively, thereby enhancing model performance. Despite the common use of Feedforward Neural Networks (FNN) in such applications, decision tree methods based on gradient descent boosting, which

are well-suited for handling large feature sets with lower computational training costs, have rarely been explored in this context. To address the issue of process drift, a live-updating transfer learning approach is implemented with FNN models. This approach periodically updates the model using up-to-date data to reduce the need for physical measurements, leveraging a base model trained on historical data. Additionally, this paper discusses and applies a data augmentation method, linear Mixup, to mitigate commonly encountered data insufficiency issues, which may arise from sensor failures or pre-mature process lines. The Mixup method has been shown to significantly improve model performance as data-adaptive regularization, offering a promising solution to these pervasive challenges [20].

This work is organized as follows: Chapter 2 provides an overview of the datasets with Chapter 2.1 going more in depth on the preprocessing procedures for both etching and slider production tools, Chapter 2.2 presents the machine learning model construction for both classification and regression tasks, Chapter 3.1 and Chapter 3.2 demonstrate and discuss the model performance results on both tools, and Chapter 4 summarizes the findings of this work.

Chapter 2

Data Preprocessing and Modeling

This section provides an overview of the solution objective, data collection, physical functionality, and specific machine tools for the use case used in this study. As mentioned, the use case encompasses five plasma etching machines at various Seagate factories used in wafer production distinct from those examined in previous research [1], and an additional seven slider tools from the production of the slider component used in hard disk drives (HDD). The 12 datasets, one from each machine, made it possible to extend the prior study on data aggregation on similar machines and processes to a study on similar machines but different processes. The solution objective remained the same as the previous study [1], in which the model was constructed to use machine tool data to determine (predict, before metrology) if the final thickness of the deposited layer PASS or FAIL. PASS refers to if the wafer is expected with high probability to fall within a specified quality range, and FAIL is if it does not. As in the previous study, a classification model was applied. In addition, a regression model was also developed for the slider production process to quantitatively predict the depth of the milling process. Both models aim to determine whether the output products of these tools are expected to meet specific metrological standards. If the performance of these models is sufficiently high, this virtual metrology application can increase the machine and staffing productivity of the physical metrology quality assurance process.

The datasets produced by the etching process facilitate the development of a binary classification model to perform fault detection. This model classifies the final thickness of the deposited layer as PASS if it falls within a predefined range and FAIL if it does not. For the slider production process, a regression model is developed to predict the depth of the milling process. This model aims to determine whether the outputs of these tools meet specific metrological standards, thus serving as an effective soft sensor for quality assurance for multiple products in manufacturing processes.

The details of the data collection and description of the tools and modules of datasets were elaborated in a prior work [1] and demonstrated in Figure 2.1. In summary, the manufacturing process requires raw wafers to undergo a sequence of process steps to transform into complete products. Due to the repetitiveness of these processes, a wafer may interact with the same kind of tool multiple times at different processes. This work analyzes etching and slider production tools, where each tool functions as a reactor and contains modules that operate as nearly independent reaction chambers to conduct specific processes. Tool-module combinations are designated in the T-PM format; for instance, 'T1-PM1' refers to module 1 in tool 1. For the rest of the paper, a data point is defined as a data vector of all features after averaging across the duration of the batch. A dataset is the collection of data points obtained from a specific tool-module combination.

2.1 Data Generation and Preprocessing for Etching and Slider Tools

2.1.1 Wafer Production Etch Tools

Operational data from each wafer production etch tool was collected and consistently organized by the CESMII Information Model or Profile pictured in Figure 2.1. The data from each machine comprised two data types: 32 streamed, time-based numerical sensor measurements and 2 cate-

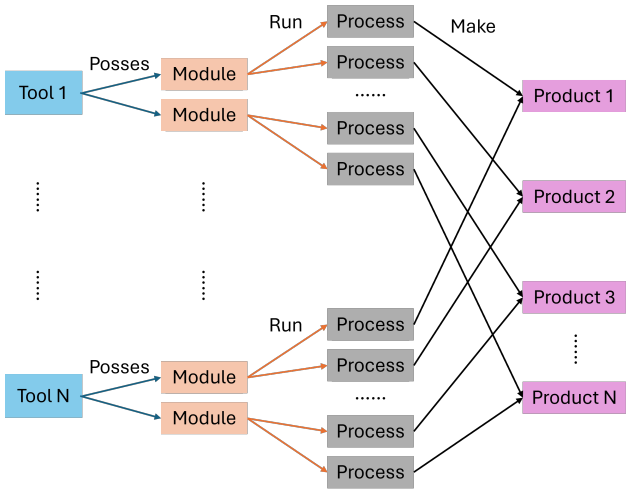

Figure 2.1: The industrial etching equipment's manufacturing system consists of multiple etching reactors, each equipped with several modules capable of running different processes. The production begins with pure silicon wafers, which undergo a series of processing steps before transforming into the final product.

gorical features. The streamed, numerical data, sourced from various physical sensors, include operational measurements such as pressure and gas flow, i.e., state variables that can be measured directly and in real-time. Categorical features describe process and material specifics that are decisions about the use of the machine. The two features included are an alphanumeric process ID that captures the "recipe" of machine functions for a particular product and the substrate family ID. Each family is a finite set of classes. The steamed operational data was averaged for each variable over duration of a batch run which is typically on the order of 20 minutes. Data collected cover five years of operation, from February 2018 to December 2022. The five etch tool datasets are categorized by machine number and process module (single machines that have parallel etch modules) as T51-PM1, T52-PM1, T52-PM2, T53-PM1, and T53-PM2.

Again, the solution objective is to build a binary prediction model by mapping these operating data to PASS and FAIL outcomes. Data from 2018 through 2021 are allocated for training, validation and the tuning of the model hyperparameters. The 2022 data, the most recently collected,

were reserved for testing. Selecting data from the oldest rather than the newer sets and then testing with the most recent data ensures that model effectiveness with respect to "normal" operational shifts and machine changes over time are accounted for and evaluated in the context of current and future production scenarios.

In addition to data selection, data preprocessing is also crucial for preparing datasets for model training process. Prior studies [21, 22] have highlighted that preprocessing can significantly impact model training performance, underscoring the importance of selecting those preprocessing methods that optimize model performance for an operational situation [23]. In this work, preprocessing involves removing or padding invalid/null data points, encoding discrete variables, and normalizing numerical data points for maximum performance accurately. The workflows for preprocessing inputs and outputs are graphically illustrated in Figure 2.2 (input data) and Figure 2.3 (outputs).

With reference to Figure 1, the first step is remove invalid data. An invalid data point is defined when critical sensor data are absent, such as the output measurements of oxide thickness, the pass/fail status, or when a substantial number of the input sensor data is missing. Missing data points can be attributed to the temporary malfunction of sensors within a batch run, not uncommon with industrial data. In scenarios where extensive numbers of sensor data are missing (usually over half of the features), zeroing all missing features or feature paddings could detrimentally affect the dataset's integrity and, consequently, the model training performance. For sensor data points in which a small number of the measurements (usually only one or two features) are missing, these missing values are replaced by the average value of the remaining data points in the data set. This imputation method helps maintain the consistency and reliability of the dataset, ensuring that the training process is not skewed by gaps in the data. This strategy of handling missing data preserves the underlying statistical relationships and prevents the introduction of bias that could mislead the learning algorithm. We note that the sensor data is collected as time-series measurements and averaged over the duration of each processing step to generate representative values for various features. Once this data is properly recorded, it is treated as accurate and reliable, as the sensors

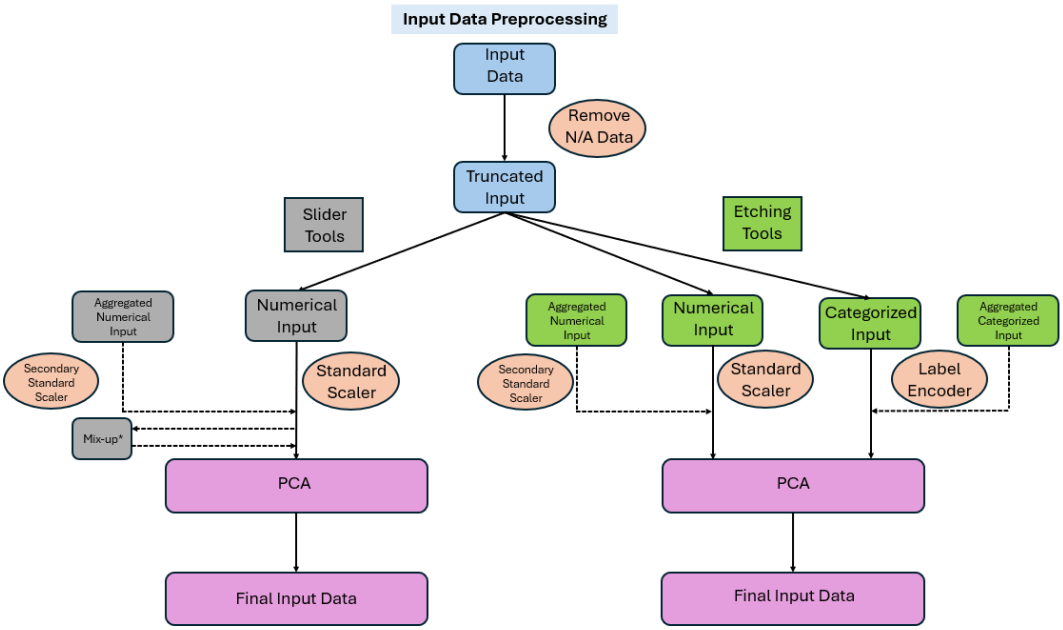

Figure 2.2: This figure illustrates the input data preprocessing workflow for etching tools and slider production tools. Common steps include invalid data removal, normalization, and dimension reduction. Etching tools incorporate encoders for categorical data management, whereas slider tools utilize Mixup to mitigate data scarcity.

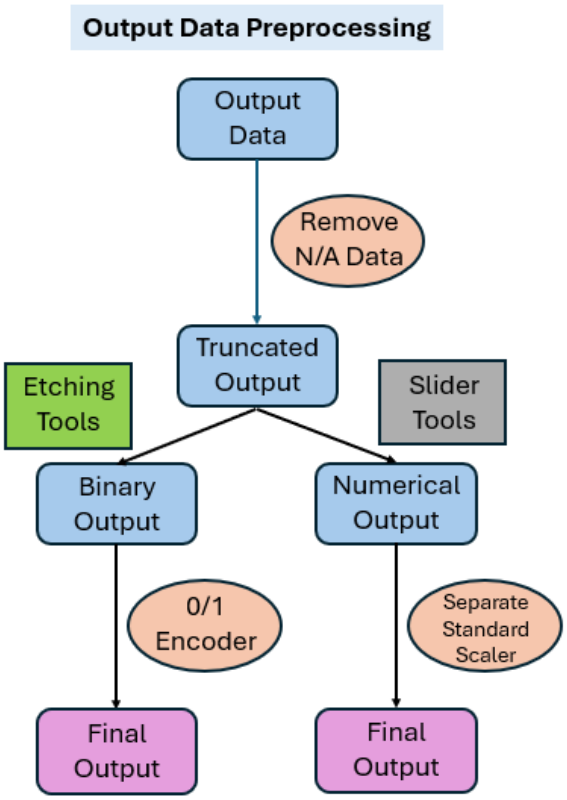

Figure 2.3: This figure illustrates the output data preprocessing workflow for etching tools and slider production tools. Common steps include invalid data removal, etching tools requires 0/1 encoding, and slider tools requires separate scaling.

on the machine have been qualified by operations to ensure proper functionality and measurement accuracy.

Conversion of categorical features into numerical formats is essential for their integration into numeric-based machine learning models. Each categorical feature, such as the process ID and the substrate family ID, must be uniquely encoded using methods such as label encoding or one-hot encoding. In this work, label encoding was used to convert process name IDs and substrate family IDs into numerical values. To ensure consistency across different machines, substrates, and production processes, the encoding was created using all available datasets combined, covering all possible entries. This approach prevents issues that could arise if the encoder were trained on a limited dataset. For example, if a tool or module only runs certain processes, an encoder trained only on that data might fail when applied to another tool with different processes, leading to errors due to missing values. By including all possible entries during encoding, we ensure that the data remains compatible across different tools and processes. For the output variable, which is either PASS or FAIL condition, binary encoding scheme is adopted, where '1' represents a PASS and '0' denotes a FAIL.

Normalization and scaling of data are critical for optimal training performance, when the input features span wide numeric ranges and scales. For instance, in this use case some features might range from 0 to 1, while others range from 0 to 100,000. Normalization was critical to avoiding gradient vanishing or gradient explosion [24]. Although gradient-boosted, tree-based methods like XGBoost and AdaBoost are inherently less sensitive to scale differences due to their reliance on decision tree structures, normalization still plays a significant role. Unscaled data can introduce inefficiencies in node splitting such that features with larger value ranges can dominate the feature selection in split decisions, potentially leading to suboptimal splits that do not capture the true underlying patterns in the data effectively.

To ensure all features are normalized consistently across different datasets, a standard scaler is applied to each dataset independently, normalizing the features to a mean value of 0 and a standard

deviation of 1. This scaling process adheres to the formula:

$$Z = \frac{X - u}{s} \tag{2.1}$$

where Z is the output vector of a scaled numerical feature, X is the input vector of the original feature, u is the average value of X, and s is the standard deviation of X. In the context of data aggregation, a method detailed in a previous work [1], where model training involves combining multiple datasets to improve the model performance by increasing data variability and data volume, it is found to be beneficial to scale each dataset separately prior to concatenating datasets. In other words, the concatenations should happen between scaled datasets but not raw data. This method is necessary because, within each module's reaction chamber, the distribution of features often varies significantly. For example, although the same feature across different datasets may be similar in scale, direct scaling of the aggregated dataset could distort the inherent distributions of each dataset if the differences in scale and standard deviation are substantial. As shown in Table 2.1, the mean percentage difference of average values across all features exceeds 40% between most tools, indicating significant distribution shifts between datasets. This distribution distortion can mislead the model if the datasets are concatenated before normalization, then impacting its performance. Consequently, separate scaling for each dual-tool combination is implemented to maintain the integrity of their original distributions relative to the output. This approach preserves crucial relationships within the data, ensuring more reliable and accurate model training outcomes.

Table 2.1: Mean Percentage Difference of Average Value of All Features

	T51-PM1	**T51-PM2**	**T51-PM3**	**T52-PM2**	**T53-PM2**
T51-PM1	N/A	64%	14%	90%	43%
T51-PM2	77%	N/A	75%	55%	46%
T51-PM3	20%	86%	N/A	101%	66%
T52-PM2	5865%	3208%	5274%	N/A	9932%
T53-PM2	47%	45%	45%	81%	N/A

After the normalization and aggregation process, dimension reduction is facilitated using Principal Component Analysis (PCA). It is worth noting that PCA requires the data to be normalized before its application. Normalization standardizes the range of features, ensuring that each feature contributes equally to the analysis and that features with larger scales do not dominate the principal components. This prerequisite is crucial because PCA is sensitive to any variance in the initial variables. The PCA process begins by calculating the covariance matrix for the data, which identifies the directions in which the data varies the most. If the data are not normalized, features with higher absolute values could disproportionately influence the covariance matrix, leading to biased principal components that do not accurately reflect the underlying data structure. Subsequently, the eigenvalues and eigenvectors of this covariance matrix are computed. The eigenvalues are sorted in descending order, and their corresponding eigenvectors are aligned accordingly. The principal components are selected based on these sorted eigenvalues, those that explain the most variance are retained (in this work 99.9% of variances are retained), discarding the less significant components (non-contributing features, noise). This selection is critical because features that exhibit higher variance are considered more influential for the model's predictive accuracy. The calculation process is shown below:

$$Q = X^T X = W \Lambda W^T \tag{2.2}$$

where X represents the data matrix with dimension $(k \times d)$, where k denotes the number of data points and d represents dimensions of datasets. Q refers to the covariance matrix, which is a square matrix of dimension $(d \times d)$. Λ is a diagonal matrix consisting of eigenvalues of Q arranged in descending order. Correspondingly, W is a matrix of the eigenvectors of Q, aligned in the same sequence as the eigenvalues in Λ. The linear transformation from the original data space to the

principal components space is represented by equation below:

$$T_L = XW_L \tag{2.3}$$

where L is the number of reduced dimensions determined by the proportion of variance to be retained, T_L represents the transformed data matrix with dimension $(k \times L)$, and W_L is the first L columns of W matrix.

Each principal component is a linear combination of the original features, representing a new direction in the feature space. By focusing on these principal components, PCA effectively reduces the dimensionality of the data. This reduction not only compresses the data with minimum loss by emphasizing directions with most significant variations, but also helps in mitigating the risk of overfitting by reducing feature numbers and noise in the data. In this work, 99.9% of variances are retained, and the resulting feature space has dimension between 10-15, corresponding to 60% to 75% reduction in dimension.

2.1.2 Slider Production Tools

Slider tools are a different part of the manufacturing process from wafer production where machine tools are used to make specific parts used in the production of data storage devices. For this study, we focused on a milling machine step which is used to establish a critical surface depth for the slider component. The modeling objective is the same virtual metrology solution for wafer production in that milling machine sensor data were collected, contextualized, processed and engineered for building a model to predict milling depth. In this study we have benchmarked data processing methods and workflow, and we compare and contrast the approach to building the virtual metrology solution for two different production applications. As with wafer production, modeling begins with data analysis. Since the goal is to predict milling depth, a continuous variable, a regression model is developed. This model leverages multiple features, approximately 20 in total, to estimate milling

depth with precision. However, an important observation from the data is that milling depth values tend to cluster into discrete ranges, corresponding to specific product specifications. Figure 2.4 illustrates these clusters, with milling depths predominantly falling within groups of 1450–1550, 1650–1850, and 2150–2350. These clusters reflect the limited variety of products, each with its own depth requirement, and are an inherent characteristic of the manufacturing process.

Given this natural grouping, the regression output can be easily mapped into predefined depth ranges, allowing for a multi-class classification approach. Instead of building a separate classification model, the regression predictions are assigned to the nearest cluster, effectively classifying each wafer's milling depth into a corresponding product category. Additionally, this classification approach aids in identifying failures. Failures are defined as wafers whose milling depth deviates significantly from the expected clusters for a given tool-module combination. For example, if a particular tool is designed to produce only one type of product, any wafer with a milling depth outside its designated range is considered a failure, even if its depth aligns with a valid range for another product. In this context, failures indicate deviations in production quality, highlighting potential process anomalies. This depth-failure classification enhances both the model's practical utility and its ability to detect defects in real-world manufacturing.

For classification, data points outside of the major depth ranges are flagged as failures, with the distribution details of these principal ranges cataloged in Table 2.2. The datasets relevant to slider production, namely T07-PM1, T07-PM2, T15-PM1, T02-PM1, T02-PM2, T01-PM1, and T05-PM2, consist of 20 features each, similar to the datasets from etching tools but encompassing fewer features and data points. The organization of data points within major depth categories for each dataset is also documented in Table 2.2, where red numbers mean data points in outlier regions.

Data preprocessing for these datasets follows a similar approach to that used for etching tools, including the removal of null values, padding of missing data, normalization, aggregation, and dimension reduction. However, since the slider production datasets do not contain categorical

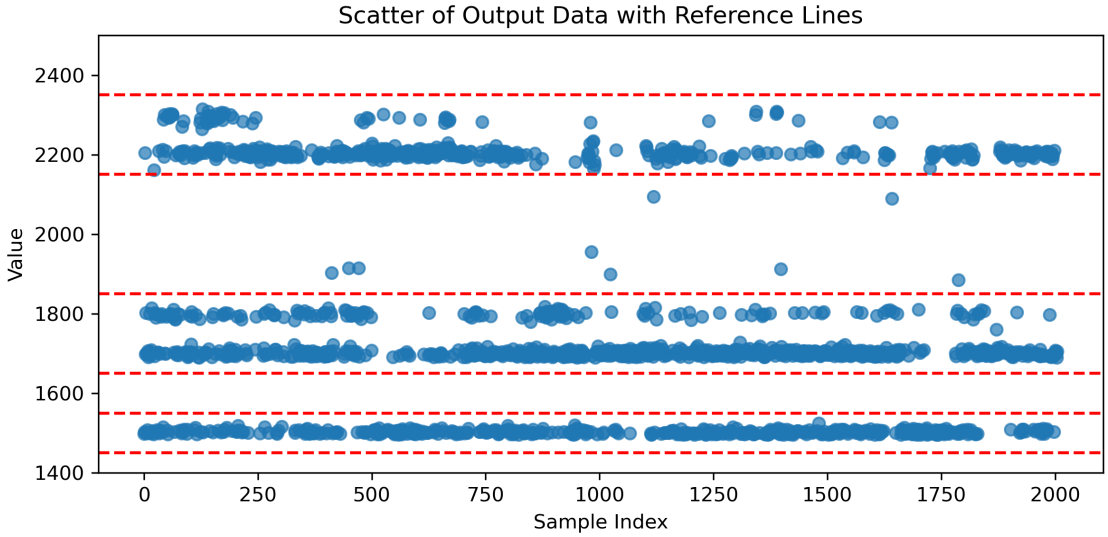

Figure 2.4: Scatter plot of T15-PM1 output, illustrating how data points are primarily grouped into several clusters.

Table 2.2: Data Distribution Across Different Ranges

	1450–1550	**1650–1850**	**2150–2350**	**Others**	**Total**
T07-PM1	1	869	9	22	901
T07-PM2	7	992	4	11	1014
T15-PM1	518	943	530	15	2006
T01-PM1	98	198	158	11	465
T02-PM1	0	742	108	24	874
T02-PM2	1	685	18	24	728
T05-PM2	0	153	0	3	156

features, the encoding step is not required. As shown in Figure 2.2, unique to slider tools, however, is the application of separate scaling on the output data during aggregation, as the outputs of etching tools datasets are binary values. In this aggregation process, one dataset is designated as the primary or 'major' dataset aimed at performance enhancement, while other datasets serve as supplementary or 'helper' datasets. The output scaler is tailored to the distribution of the major dataset before being applied to the combined dataset to preserve its integrity. This strategy is crucial because using a universal scaler across the aggregated data could distort the distribution of the major dataset of interest, such as the single-region T07-PM1 versus the three-region T15-PM1.

After normalization and aggregation, PCA is applied to the input data to retain at least 99.9% of the original variance, significantly reducing dimensionality while preserving essential information. The number of principal components retained varies by dataset, typically resulting in about 10 principal components, effectively halving the original feature set. This dimensionality reduction is critical in managing the complexity and enhancing the interpretability of the models developed from these high-variance datasets.

Due to the limited data availability for slider production tools as characterized by a shorter data collection period of only one year compared to five years for etching tools and a lower frequency of data recording, enhancements in model performance are necessary. To address this, a data augmentation technique known as Mixup was employed to create artificial data points by interpolation between existing data points.

However, given that the slider data belongs to several discrete regions, interpolating data points between major regions was avoided since doing so could lead to non-existing milling depth. For data aggregation, each dataset undergoes Mixup independently within its respective major regions, and the resulting post-processed datasets are then concatenated. This ensures that the augmented data remains representative of the true distribution patterns observed in the production environment. To evaluate the impact of different interpolation strategies on model effectiveness, two distinct Mixup schemes were tested. The first scheme is tri-point interpolation, which creates

two artificial data points between each two data points. This scheme is formulated as follows:

$$\begin{aligned} x_{mix} &= 0.33x_i + 0.67x_{i+1},\ x_{mix} = 0.67x_i + 0.33x_{i+1} \\ y_{mix} &= 0.33y_i + 0.67y_{i+1},\ y_{mix} = 0.67y_i + 0.33y_{i+1} \end{aligned} \tag{2.4}$$

The second scheme creates one data point directly in the middle of two data points as formulated as follows:

$$\begin{aligned} x_{mix} &= 0.5x_i + 0.5x_{i+1} \\ y_{mix} &= 0.5y_i + 0.5y_{i+1} \end{aligned} \tag{2.5}$$

Beyond data augmentation, Mixup also acts as a form of regularization, effectively smoothing the decision boundaries of the model. This smoothing is achieved by encouraging the model to perform linear interpolations between features and their associated targets in the input space, which can reduce the model's confidence in far-reaching predictions. Such a characteristic is particularly useful in mitigating overfitting, as it prevents the model from learning overly complex patterns that are heavily dependent on the specific training data distribution. Instead, Mixup encourages the model to generalize better to new, unseen data by promoting a broader exploration of the feature space and reducing the likelihood of drastic output changes in response to small variations in input. This regularization effect makes Mixup a valuable technique in enhancing the robustness and generalizability of machine learning models [20].

2.2 Model Training

2.2.1 Introduction to Machine Learning Models

Feedforward Neural Networks (FNNs) are widely used for modeling processes involving regression and classification under large amounts of data and features. However, tree-based ensemble methods like Extreme Gradient Boosting (XGBoost) demonstrate advantages when dealing with

high-dimensional feature spaces and significant noise levels. This is because:

- FNNs require careful normalization, can consume more computational resources, and may suffer from overfitting in complex architectures.
- Tree-based approaches (like XGBoost) require less stringent normalization, often train faster on moderate size dataset, and are generally robust to overfitting due to inherent regularization.

In industrial applications, datasets often present challenges such as high-dimensional feature spaces, missing values, class imbalance, and the presence of significant noise. These characteristics make tree-based approaches particularly well-suited for industrial data modeling. The key advantages include:

- **Robustness to Noisy and Incomplete Data:** Industrial datasets frequently contain sensor readings, production metrics, and operational parameters that may be affected by measurement errors, environmental conditions, or missing entries. Tree-based approaches handles such inconsistencies effectively using its optimized split criteria and missing value handling mechanisms, allowing it to make robust predictions despite imperfect data.
- **Feature Selection and Interpretability:** Unlike FNNs, which often require careful feature engineering and are perceived as black-box models, tree-based approaches naturally identifies the most important features during training. This ability is critical in industrial settings where engineers and decision-makers need interpretable insights for process optimization, fault diagnosis, and predictive maintenance.
- **Computational Efficiency and Scalability:** Industrial datasets in this study range from 10,000 to 30,000 data points. Training deep neural networks on such data is more computationally expensive than XGBoost, even with a GPU, and require extensive hyperparameter tuning. Tree-based approaches, on the other hand, is optimized for speed and scalability for

mid size datasets as in this work, leveraging parallel processing and tree-pruning techniques to efficiently handle large datasets with minimal computational overhead.

- **Regularization for Generalization:** FNNs, particularly deep architectures, require careful tuning of dropout rates and batch normalization to prevent overfitting. Tree-based approaches can incorporate built-in regularization techniques such as L1 (LASSO) and L2 (Ridge) penalties, along with shrinkage (learning rate adjustment), ensuring better generalization performance without extensive fine-tuning.

Given these advantages, tree-based approaches emerges as a highly effective tool for modeling industrial processes, where robustness, interpretability, and computational efficiency are paramount. XGBoost is one example of a tree-based ensemble method combining the predictions of multiple decision trees in a gradient-boosted framework. The algorithm begins by training a base tree, then computes the residual (the difference between the current prediction and the true label). Subsequent trees are trained to optimize these residuals. The final prediction for sample x_i after t trees is:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta \cdot f_t(x_i), \tag{2.6}$$

where η is the learning rate and f_t is the t-th decision tree. This iterative approach refines the model incrementally, making XGBoost particularly effective in many structured data tasks, which are tasks that involve datasets where features are well-defined, typically organized in tabular format, and consist of numerical, categorical, or ordinal variables. These tasks are common in industrial settings, where data is collected from sensors, production logs, and quality control systems.

2.2.2 Model Training on Etching and Slider Tools

Etch tool data are used to train a ML classification model to perform a binary prediction of PASS or FAIL product at the completion of the machine tool process step. Because the FAIL rate from the

process step is small($\sim 2\%$), the data are highly imbalanced between PASS and FAIL. We therefore use a weighted cross entropy loss approach to emphasize FAIL conditions. Cross-entropy loss, also known as log loss, is a commonly used loss function for classification tasks. It measures the divergence between the true labels and the predicted probabilities, penalizing incorrect predictions more heavily. In binary classification, it is defined as:

$$L(y_i, \hat{y}_i) = -w_0(1 - y_i)\log(1 - \hat{y}_i) \; - \; w_1 y_i \log(\hat{y}_i), \tag{2.7}$$

where w_0 and w_1 are class weights set as:

$$w_0 = \frac{1}{n_0}, \quad w_1 = \frac{1}{n_1}, \tag{2.8}$$

and n_0 and n_1 denote the number of PASS and FAIL samples, respectively. It's worth mentioning that in our datasets FAILs are usually represented as 1 instead of 0. This scheme emphasizes the minority class to improve the detection of FAIL events.

80% of the etch tooling dataset (2018–2021) was used for training, while the remaining 20% was separated out for validation. To systematically tune hyperparameters of the machine learning model, we performed a 5-fold cross-validation on the training data. Cross-validation is a technique used to assess the generalizability of a model by splitting the dataset into multiple subsets. In 5-fold cross-validation, the training set is divided into five equally sized folds, where the model is trained on four folds and validated on the remaining fold. This process is repeated five times, with each fold serving as the validation set once. The final model performance is averaged over all iterations, reducing the risk of overfitting and improving robustness.

To further optimize hyperparameters, we apply a grid search exploring various learning rates, the number of estimators, and regularization strengths for both FNN and XGBoost. Grid search systematically evaluates predefined combinations of hyperparameter values to identify the optimal set that maximizes model performance. This approach ensures that different configurations are

tested comprehensively, balancing model complexity and predictive accuracy. The final selected hyperparameters are shown in table 2.3 and table 2.4, respectively, where the underlined parameters represent the chosen optimal values.

Table 2.3: Hyperparameters for FNN

Hyperparameter	**Range/Candidate Values**
Number of layers	[1, 2, 3]
Number of neurons	[16, 32, 64]
Activation function	['relu', 'sigmoid']
Dropout ratio	[0.0, 0.2, 0.5]
L2 regularization	[0, 10^{-4}, 10^{-3}, 10^{-2}]
Training epochs	[500, 1000, 2000]
Early stops	['Yes', 'No']

Table 2.4: Hyperparameters for XGBoost

Hyperparameter	**Range/Candidate Values**
Learning Rate	[0.1, 0.3, 0.5]
Max Tree Depth	[6, 8, MAX*]
Subsample Rate	[0, 0.5, 1]
L1 Regularization	[0, 1, 10]
L2 Regularization	[0, 1, 10, 100]

* Dimension after PCA

In all kinds of machine operation that run continuously over a long periods of time, process drift poses a significant challenge to maintain accurate model performance over time, as even the sensors themselves can drift and develop bias. Process drift occurs due to various factors such as equipment aging, accumulation of unwanted deposits on machinery, and other operational changes that alter system behavior with time. For example, in etch machines, process drift can manifest as

a gradual reduction in etch rate due to the accumulation of impurities on chamber walls[25]. This buildup alters plasma conditions, leading to deviations in feature dimensions and film thicknesses over time. If not accounted for, such drift can degrade the predictive performance of machine learning models trained on historical data, necessitating periodic model recalibration.

As a result, models trained on initial datasets may gradually lose accuracy. Transfer learning addresses this by adapting existing models to new data without full retraining, preserving prior knowledge while integrating new information. This is especially valuable in industrial settings, where collecting sufficient data in different environments is costly due to the need for physical measurements. Additionally, in scenarios such as data drift, continuous updates are required, but new data may never be sufficient for training from scratch before being outdated.

Our implementation of transfer learning followed a three-step procedure designed to improve model performance while minimizing the need for costly physical measurements:

1. **Base Model Training:** The procedure begins with developing a base model by training the feedforward neural network (FNN) on old data, which in our case is the data from 2018-2021, to establish foundational performance metrics. This base model serves as the starting point for all subsequent updates and evaluations.

2. **Incremental Model Updating with SGD:** Once the base model is established, it is incrementally retrained on new data using stochastic gradient descent (SGD). The training data window size (i.e., mini-batch size) is a tunable hyperparameter, with examples including mini-batches of 1 or 10 data points, referred to as 'train data length' in Table 2.5. To prevent data leakage, each new data point is first evaluated using the original, pre-updated model before being used for training. Although the true label is available, the model is tested as if it had not yet seen this data point. Only after recording this evaluation is the model updated with the new data using SGD. This ensures an unbiased assessment of how well the model adapts over time while accurately measuring the effectiveness of the updating algorithm.

The use of SGD with a carefully selected learning rate enables efficient updates, striking a balance between adapting to new information and preserving existing knowledge to prevent model destabilization.

3. **Model Evaluation on Test Data:** After the model is updated in the previous stage, it is now evaluated on new test data points. At this stage, the model is not updated—only its performance is measured. The ratio between the number of test data points and the training data points from the previous step is defined as the 'Test/Train Length Ratio' in Table 2.5. This evaluation step ensures that the model can make accurate predictions on new data without additional retraining. The goal of transfer learning here is to improve model performance while minimizing the need for physical measurements, which are costly and time-consuming. If the model were retrained on every new data point, it would require collecting new labels through physical measurements, defeating the purpose of transfer learning. Although all data in this work are labeled through physical measurements, the labels are used solely to evaluate the effectiveness of transfer learning. From the model's perspective, it does not have prior knowledge of the labels during training. By testing the transfer learning scheme in this controlled setting, its effectiveness can be validated before future real-world applications. After completing this evaluation step, the process returns to the previous stage, where the model is updated with the next batch of new data points, continuing the learning cycle. After completing Step 3, the process returns to Step 2, where the model is updated with the next batch of new data points. This continuous cycle of evaluation and updating is known as real-time update, allowing the model to adapt dynamically to evolving data patterns.

Hyperparameters such as the learning rate, training data length, and test/train length ratio are optimized to achieve robust model performance. This optimization is conducted through a systematic grid search, as detailed in Table 2.5. For each tool, all combinations of train data length and train/test length ratio are thoroughly tested to examine the effectiveness of transfer learning.

Table 2.5: Hyperparameters for Transfer Learning Model Training

Parameter	Hyperparameters
Learning Rate	[0.0001, 0.001, 0.01]
Training Data Length	[1, 10]
Test/Train Length Ratio	[1, 2, 4]

In summary, transfer learning efficiently maintains model accuracy amid process drift by leveraging knowledge from historical data and incorporating incremental updates. This approach sustains performance while reducing the costs of full retraining, making it well suited for evolving industrial processes and the practical constraints of manufacturing environments.

For the slider datasets, the task is a regression problem instead of binary classification. Therefore, the Mean Squared Error (MSE) is employed as the primary loss function:

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

where $\hat{y}_i$ and y_i are the predicted and actual values for sample i, and N is the total number of samples.

Due to the limited dataset size (collected over a shorter time period), fewer layers and neurons are used in the FNN to prevent overfitting. XGBoost, however, retains its standard set of hyperparameters, exploiting its tree-based mechanism to handle even relatively small datasets. To prevent overly small partitions, we use an 80%–20% train/validation split without a separate time-based test set. Given the limited data volume and its confinement to a one-year range, time-based prediction is less critical.

By applying FNN and XGBoost with these tailored hyperparameters and the MSE loss, we achieve reliable regression performance within the constraints of a smaller dataset. Given the limited data, the FNN architecture is kept simple with fewer neurons to avoid overfitting and reduce computational costs.

Table 2.6: Hyperparameters for FNN

Hyperparameter	**Range/Candidate Values**
Number of layers	[1*, 2, 3]
Number of neurons	[8, 16, 32]
Activation function	['relu', 'tanh']
Dropout ratio	[0.0, 0.2, 0.5]
L2 regularization	[0, 10^{-4}, 10^{-3}, 10^{-2}]
Training epochs	[500, 1000, 2000]
Early stops	['Yes', 'No']

* Single Training

Chapter 3

Results and Discussion

3.1 Etching Tools

The performance of the machine learning models introduced in Chapter 2.2.2 that were trained on industrial data we received are best evaluated within the bounds of the specific machines, operational environments, and conditions in which they will be applied. As discussed in a previous study, the etch machines in this study perform well with low percentages of product FAIL rates; therefore, the data distribution is highly imbalanced toward PASS vs. FAIL runs [1]. There are four possible outcomes in a binary classification:

1. **True Positive**: Correctly identifying a PASS.
2. **False Negative**: Incorrectly labeling a PASS as a FAIL.
3. **True Negative**: Accurately detecting a FAIL.
4. **False Positive**: Incorrectly labeling a FAIL as a PASS.

True positives and true negatives indicate that the classification model is correctly qualifying the product state. False negatives, though undesirable, can be mitigated since all flagged failures

undergo an additional metrology inspection where false negatives can be corrected by manual measurements. The larger concern in these operations are false positives, as they are a much more expensive miscategorization compared to false negatives. False positives are more costly because downstream operational resources continue to process the defective product, wasting resources and time. In most cases, false positives will eventually be identified through later metrology steps or final product reliability testing [26].

An acceptable classification model for this operating situation needs to have sufficient accuracy and precision in identifying true FAIL and PASS results but also in minimizing false positives to reduce the cost of classification errors. In operational terms, this virtual metrology solution leads to significant economic benefit by debottlenecking the physical metrology step and increasing production volume.

The performance metrics for the classification models must reflect the ability of a model to control its false positive rate. Performance is adjusted by modifying classification thresholds that affect the balance among all four possible classifications. Increasing the sensitivity to misclassified wafers reduces false positives but raises false negatives and vice versa. For this application there is a need to demonstrate the ability to tune the model by considering performance across a range of threshold settings. Confusion matrices, while commonly used to evaluate classifier performance, only address results at a specific threshold [27]. They fail to capture overall model performance for various settings. Accuracy evaluation is unsuitable because of the extreme PASS vs. FAIL imbalance. For instance, a model that always predicts "PASS" may appears highly accurate because the vast majority of classifications are PASS, but is functionally useless with a 100% false positive rate.

Each model does not have a single, fixed false positive rate; rather, its performance can be adjusted by modifying classification thresholds. Increasing sensitivity to misprocessed wafers reduces false positives but raises false negatives, whereas a more lenient approach does the opposite. Confusion matrices, while commonly used to evaluate classifier performance, only depict results

at a specific threshold and fail to capture overall model capability across various settings. Other common evaluation metrics, like accuracy, are also unsuitable due to the extreme class imbalance in manufacturing datasets. For instance, a model that always predicts "PASS" may appear highly accurate but is functionally useless with a 100% false positive rate.

The Receiver Operating Characteristic (ROC) analysis is a better analysis tool for this use case because it directly plots the true positive rate (TPR) and false positive rate (FPR) across the entire span of thresholds. The output of the classifier produces continuous scores in the range [0,1] of the classification threshold. The default threshold is of 0.5, which is equivalent to guessing an outcome. We therefore want to be able to adjust this threshold to get an acceptable performance. Setting the threshold to 0 results in a TPR of 100% and an FPR of 100% (always PASS), while a threshold set at 1 results in TPR and FPR rates of 0% (always FAIL). The optimal threshold therefore requires balancing sensitivity (TPR) angainst false alarms (FPR). To objectively compare the performance of models, the Area Under the ROC Curve (AUC) is used since it is a metric that quantifies overall performance across all threshold settings when plotting TPR against FPR. A perfect model would achieve an AUC of 1, meaning 100% sensitivity with no false positives. In contrast, a model making random predictions would yield an AUC of 0.5. Since manufacturing data is typically imbalanced, ROC-AUC provides a reliable measure of classification effectiveness [28].

3.1.1 Single and Dual Tool Training

The AUC scores of the FNN and XGBoost models on each of the five etching machine datasets with single-tool (trained only on one dataset) and dual-tool (trained on the aggregation of two unique datasets, the aggregated dataset is picked by best available score) training are shown in Figure 3.1:

For single-tool training, we observed an increase in AUC scores for 3 out of the 5 tools when using the XGBoost model compared to the FNN model. The largest increase was 0.09 in T53-PM1, while the largest decrease was 0.13 in T52-PM2. In dual-tool training, XGBoost demonstrated

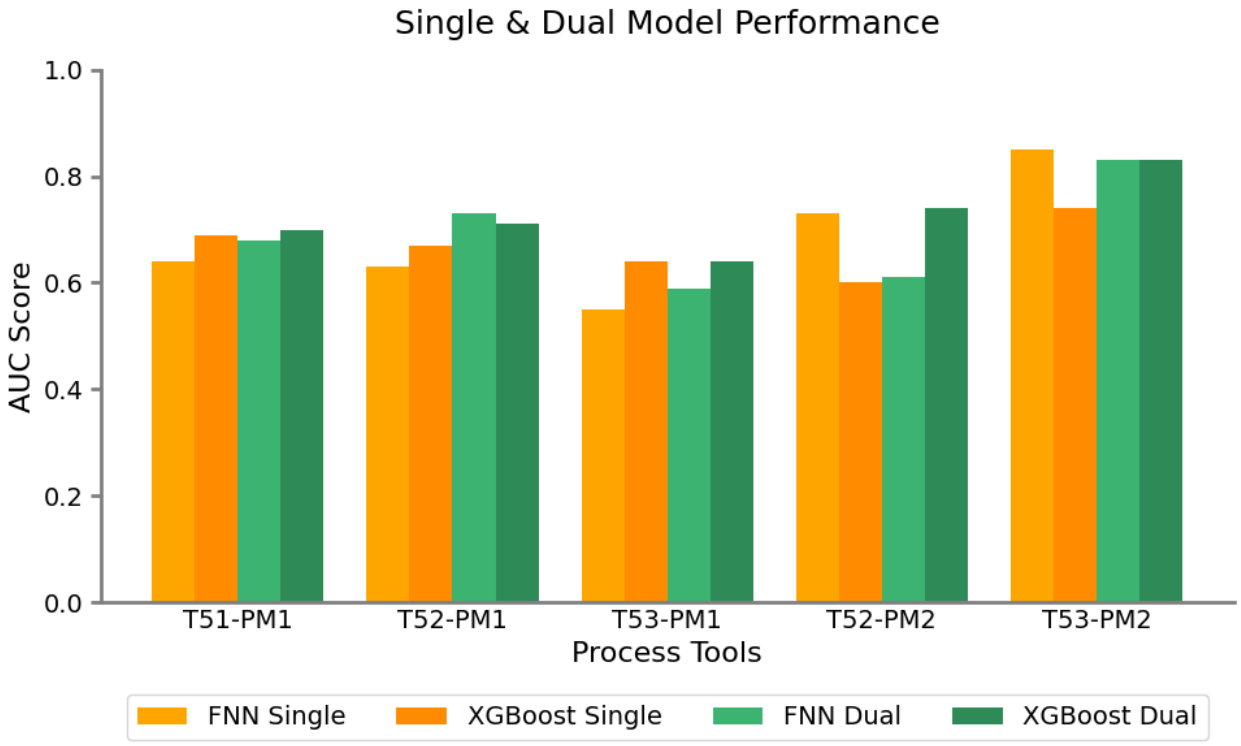

Figure 3.1: This figure shows the AUC scores for the FNN and XGBoost models applied to etching tool data using single tool and dual tool training.

better or similar performance compared to FNN across all tools. The largest AUC increase was 0.13 in T52-PM2 and 0.05 in T53-PM1, with the largest decrease being minimal at 0.02 in T52-PM1. An improvement in the ROC-AUC score indicates a better ability to distinguish between PASS and FAIL classifications. For instance, an AUC increase of 0.13 can suggest that under controlled 80% true positive rate (TPR) the false positive rate (FPR) has decreased at least 10% (the actual level depends on the specific behavior, the AUC score does not have a fixed relationship with TPR/FPR improvement) at various classification thresholds. In practical terms, this means that XGBoost reduces misclassifications that fewer failed cases are incorrectly predicted as pass, or more true failures are correctly identified. Conversely, the AUC drop of 0.13 in T52-PM2 indicates a decline in this ability, likely due to the dataset's small size. To illustrate the practical implications, an AUC score of 0.8 on T53-PM2 typically enables detection of over 90% of defective products (i.e., fulfilling the <10% FPR requirement) and reduces more than half of the associated physical measurements needed for further inspection (False negative predictions). In general, every 0.1 increment in the ROC-AUC score yields approximately 10% reduction in the required physical measurements. However, these improvements are not uniformly distributed; gains realized closer to the ideal AUC of 1.0 tend to be both more significant and more difficult to achieve.

Despite some decreases in single-tool training, the reductions were negligible in dual-tool training, and AUC scores either improved or remained stable for all datasets. This confirms that XGBoost is a competitive classifier for PASS/FAIL classification in industrial datasets. The notable fluctuations in T52-PM2 are likely due to its limited dataset size—only 863 data points compared to over 10,000 in other datasets. This small sample results in a very limited test set with just 5 fail data points, introducing bias and randomness that undermine the reliability of performance metrics for this tool. Furthermore, aggregating data across tools continues to be valuable. The dual-tool training advantage observed in prior work with FNN remains valid for XGBoost, with dual-training AUC scores consistently outperforming single-tool training across all five tools. Coupled with its superior or comparable performance, significantly lower computational resource requirements,

and easier hyperparameter tuning compared to FNN, XGBoost, along with other boosted decision tree models, emerges as a highly competitive candidate for PASS/FAIL classification in industrial applications.

3.1.2 Transfer Learning

The transfer learning results for three different train/test ratios in single-tool training are presented in Figure 3.2. As stated in Chapter 2.2, the train/test ratios (row) indicates the proportion of data points used for model updates within a given time period, the train data length (column) indicates the frequency of model updating. The heat maps for each tool are generated by plotting AUC scores for each Test/Train Ration and Train Data Length pair. A darker shade of blue indicates a higher AUC score, meaning better model performance. The heat map indicates that in most cases, variations in the train/test ratio, ranging from 1:1 to 1:4, do not significantly affect the AUC score when the train data length remains constant. The performance differences between frequent model updates (small train/test ratio) and less frequent updates (large train/test ratio) are minimal. This indicates that the model does not require persistent updates to maintain its predictive accuracy. It can be trained on a subset of the data at periodic intervals and then applied over extended periods without substantial performance degradation. This finding has practical implications. In settings where labeled data collection requires physical measurements, a reduced train/test ratio translates to fewer necessary measurements in a given time period. For example, in a 1:4 train/test ratio scenario, only 20% of the available data is used for model updates within an update period. This implies that if physical measurements are required for labeling, the measurement workload can be significantly reduced, as labeling is only needed during model training. In contrast, during the prediction phase, which accounts for 80% of the time, no physical measurements are required. In addition, in semiconductor manufacturing, where data collection is automated and continuous, an important takeaway is that effective model updates do not require all collected data. Instead, only a small subset is sufficient to maintain optimal model performance within a given time period.

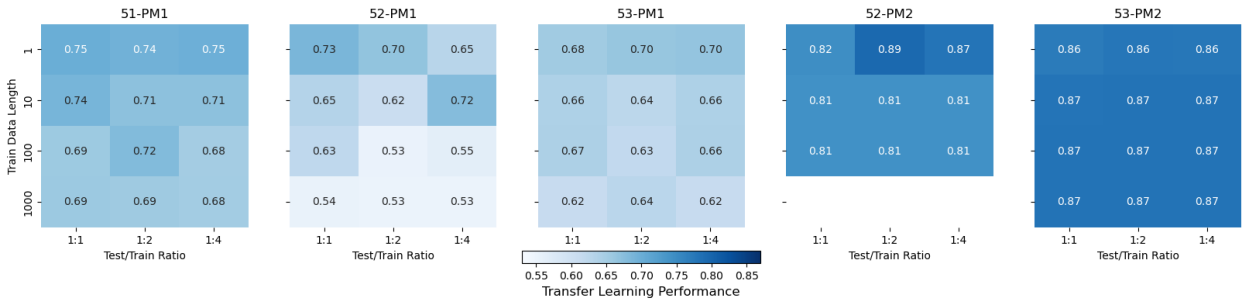

Figure 3.2: Single tool transfer learning results in heat map form. The performance difference in the same row is minimal (different train/test length ratio), but the performance difference in the same column (different train data length) is significant.

This insight allows for optimizing data usage, focusing computational resources on periodic model retraining rather than continuously processing every newly collected data point.

However, when comparing different training data lengths, a clear trend emerges: models trained with shorter datasets, which require more frequent updates in a time-series context, consistently outperform those with longer training datasets. This suggests that frequent adaptation of the model to recent data improves performance, likely because it enables the model to better capture the dynamics of the evolving system and mitigate temporal variations in the data distribution. In contrast, when the training data length is large, the model updates occur less frequently, potentially making it less responsive to shifts in the underlying data patterns. In particular, when the training data length is set to 1000 (excluding T52-PM2, which lacks sufficient data points), the performance closely resembles that of regular single tool training, indicating that transfer learning provides little to no benefit. In conclusion, the model should be updated frequently but in a low-density manner to improve the performance under process drift with maximum savings computational resources for retraining the model.

The results of transfer learning for dual tool training are presented in Figure 3.3 in the form of a heat map. Overall, the trends observed in single-tool training are largely retained, and while dual-tool training leads to slight performance improvements in certain cases, such as for T52-PM1

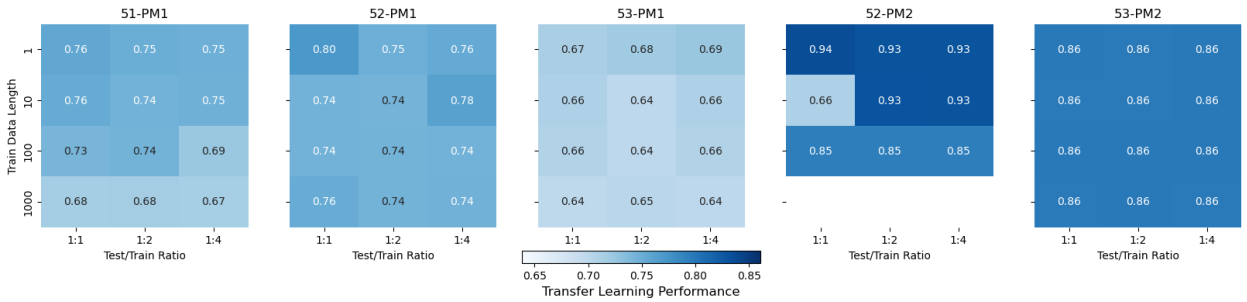

Figure 3.3: Dual tool transfer learning results in heat map form. The performance shows slight improvements compared to single-tool training and follows similar trends observed in single-tool training.

and T52-PM2, the results for most other tools remain comparable to single-tool training. This suggests that although data aggregation can enhance model robustness by exposing it to a broader range of conditions, its impact under transfer learning is limited.

One key reason for this is that transfer learning inherently allows the model to dynamically adjust to shifts in the data distribution over time. This continuous adaptation reduces the effectiveness of the generalization from aggregating because the model is already evolving to capture new patterns. Additionally, while data aggregation improves generalization across different tools, transfer learning, particularly through fine-tuning, has the opposite effect by making the model more specialized for a specific tool on the most recent dataset. As a result, the benefit of model generalization due to aggregated data may be overridden by the model's progressive adaptation to recent data. Furthermore, transfer learning itself tends to be biased towards recent data, as it incrementally modifies the model to fit the latest available information. Systematic application of transfer learning to the model is arguably the most significant reason why data aggregation is less effective .In short,since the model is continuously optimized for the most recent dataset, the contribution of earlier aggregated data diminishes over time. Consequently, while dual-tool training offers advantages in other cases, the overall benefits of data aggregation in a transfer learning framework remains negligible.

3.2 Slider Tools

The FNN and XGBoost regression models for the seven slider milling tools introduced in Chapter 2.2.2 are evaluated with two metrics: Median Absolute Error (MAE), measures the magnitude of prediction error, and median value is applied to remove the influence from outliers; and Coefficient of determination score (R^2), evaluates how model explains the variance of original dataset. Minimizing MAE and maximizing R^2 are essential for improving model performance.

The Median Absolute Error (MAE) measures the average absolute difference between the predicted and actual values. It is mathematically defined as:

$$\text{MAE} = MED(|y_i - \hat{y}_i|)$$

where y_i is the true value vector and $\hat{y}_i$ is the predicted value vector, MED is the operator that calculates the median value of a vector. Minimizing median absolute error is desirable because, unlike mean absolute error, the median is less affected by large deviations. Also, MAE can be beneficial over Mean Squared Error (MSE) because it is more robust to outliers, as it measures the median deviation rather than squaring all errors, which reduces the impact of extreme values. A smaller MAE indicates that the model's predictions are closer to the actual values on average.

The R^2 score, also known as the coefficient of determination, indicates how well a model explains the variance in the target variable. It is calculated using the following formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Here, $\bar{y}$ represents the mean of the actual values. The numerator is the sum of the squared errors, while the denominator is the total variance in the data. A higher R^2 score, approaching 1, suggests that the model accounts for a significant portion of the data's variance and it has a strong fit. R^2 is often used to compare the performance of different models, where a positive R^2 implies that

the model performs better than a model that simply predicts the mean value. Achieving a high R^2 score alone is not sufficient, because there may still be large individual prediction errors, or the model may overfit the data. Cross-validation is critical for confirming generalization capability of the model.

Balancing MAE and R^2 ensures that the model maintains both low error magnitudes and strong variance explanation. A model with low MAE and a low R^2 may fail to capture underlying patterns in the data, while a model with high R^2 but large MAE can suffer from overfitting or prediction instability.

3.2.1 Single and Dual Tool Training

Both the FNN and XGBoost models are trained similarly to the etch tools: single training, where training data only comes from the tool it is being tested on, and dual training, where training data comes from the tested tool and one of the other seven slider tools. Since the data range is primarily between 1450 and 2350, the goal is to ensure predictions remain within the controlled range and do not shift to another group. Therefore, an error below 20($\sim 1\%$) is considered small enough for effective multi-class classification, and an error below 10 (approximately 0.5%) is regarded as a near-perfect score. In the slider production tool, the mean absolute error (MAE) may be loosely compared to that of an etching tool by treating predictions within the specified major ranges as positive (true) and those outside as negative (false). Since this is a direct regression model, it is not feasible to assign multiple thresholds for classification; consequently, only a single pair of TPR and FPR can be produced, permitting indirect comparison with ROC results. In general, an MAE of around 20 corresponds to an AUC of approximately 0.70–0.75, whereas an MAE below 10 indicates an AUC exceeding 0.85. In this work, the R^2 score is less critical than the absolute error, as the primary goal is to accurately predict the group to which each data point belongs. However, R^2 remains a useful metric for assessing how well the model captures the overall variance in the dataset, providing insight into its explanatory power. The performances of both FNN and XGBoost

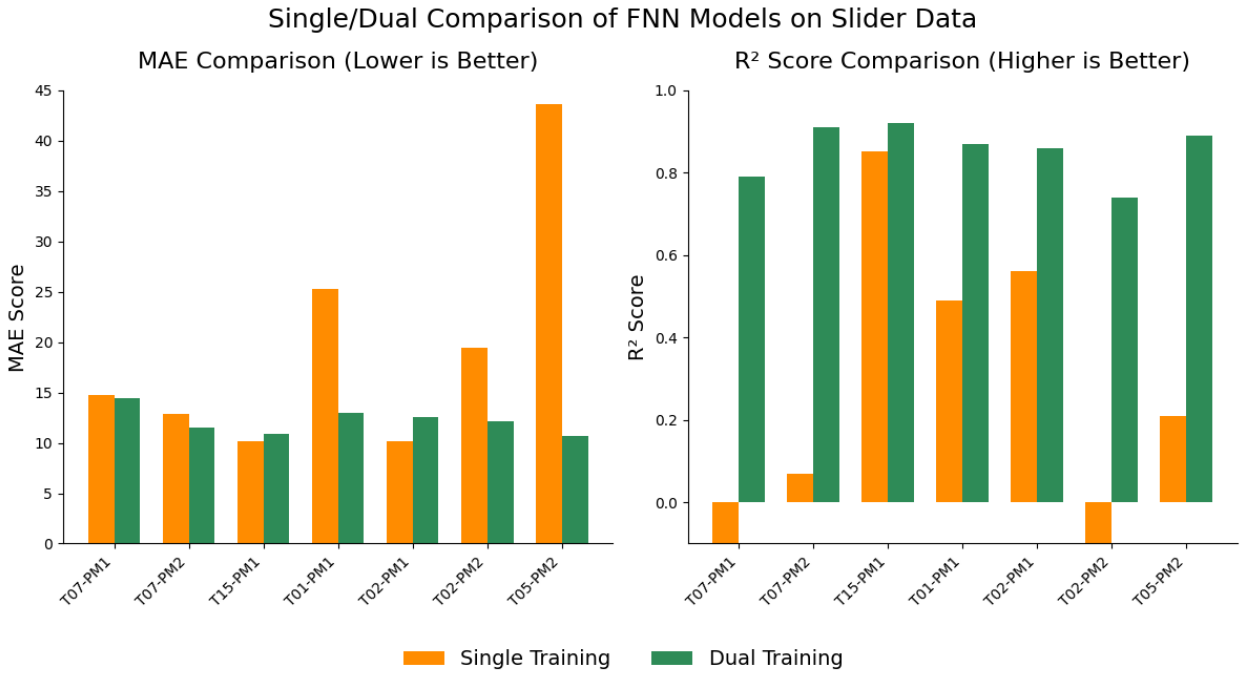

Figure 3.4: This figure shows the MAE and R^2 scores for the FNN model applied to slider tool data. MAE generally decreases when training data is aggregated with another tool. R^2 generally increase with this aggregation as well.

models using single and dual training are shown in Figure 3.4 and Figure 3.5.

For the FNN model, MAE decreased for 5 tools and increased for 2 tools after aggregation. The largest decrease was 32.85 ($\sim$ 2%), while the largest increase was 2.36 ($\sim$ 0.1%). In general, the decreases in MAE were more significant than the increases. R^2 scores increased for all 7 tools in the FNN model, with the largest increase being 1.37, change from a score below baseline to a score over 0.8.

For the XGB model, MAE decreased for 5 tools and increased for 2 tools. The decrease in MAE is not as large as seen in the FNN model but the increases in MAE was less than the one seen in the FNN model. The largest decrease in MAE was 3.86($\sim$ 0.2%) while the largest increase was 1.54($\sim$ 0.07%). Significant increase in R^2 scores are also seen in the XGB model. R^2 scores increased for 6 tools, the largest increase being 0.47 while the only decrease being -0.03. Although

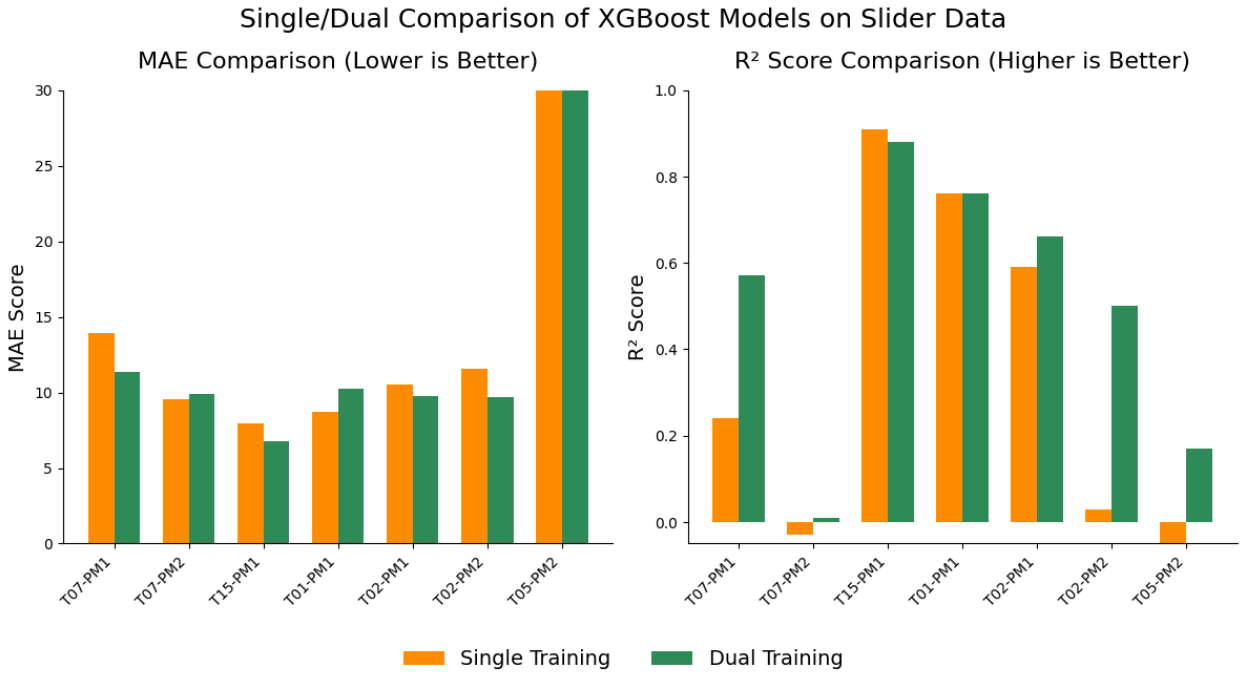

Figure 3.5: This figure shows the MAE and R^2 scores for the XGBoost model applied to slider tool data. MAE generally decreases when training data is aggregated with another tool, but not as much as seen in the FNN model. R^2 generally increase with this aggregation as well.

the decrease in MAE from dual-tool training is small for XGBoost model, the MAE score predicted by XGBoost is lower for 6 out of 7 tools than FNN.

The results demonstrate that aggregating training data from another tool improves the performance of both models in terms of reducing MAE and increasing R^2 scores. However, the magnitude of improvement varies between the two models, with FNN showing more pronounced changes in performance metrics. This indicates that FNN benefits significantly from data augmentation in most cases, resulting in greater prediction accuracy across tools. Furthermore, the R^2 scores for the FNN model improved for all seven tools, reflecting enhanced model fit and a better ability to explain variance in the target variable. The large increases in R^2, particularly for tools with low single-tool training performance, suggest that the dual training approach helps the FNN model capture important patterns that are not present when training on single-tool data alone. This improvement can be attributed to FNN's sensitivity to additional data, as the model's continuous learning structure allows it to better generalize and reduce overfitting when exposed to diverse datasets.

The XGBoost model exhibited a similar trend, with MAE reductions observed for five tools and increases for two tools, with a mean change of -1.20, which is a smaller average decrease than observed in the FNN model. This smaller decrease may be due to several factors. Firstly, XGBoost's single-tool training performance was already high, leaving less room for noticeable improvement through data aggregation. Additionally, XGBoost's tree-based architecture, with its strong regularization mechanisms and robust splitting criteria, inherently minimizes overfitting, making it less sensitive to the benefits of additional synthetic or aggregated data. On the other hand, the average increase in MAE for the two tools was 0.92, lower than that observed in the FNN model, suggesting that while the gains from data augmentation were smaller, XGBoost maintained more stable and consistent performance across different tools.

In terms of R^2 scores, XGBoost showed strong overall improvements, with increases in six out of seven tools. The only tool with a decrease experienced a minor drop of -0.03, indicating that

dual training generally enhances the model's ability to capture variance, although not as uniformly improving as observed in the FNN model. The improvements in R^2 suggest that data aggregation still contributes to better generalization for XGBoost, even though its architecture limits the extent of performance changes compared to FNN.

These findings highlight the importance of data aggregation in improving model performance in regression tasks, particularly when single-tool training data is insufficient to capture underlying patterns. While both models benefited from dual training, FNN exhibited greater performance gains, particularly in reducing MAE and enhancing R^2 scores. This difference can be attributed to the models' architectures: FNN models rely heavily on complex feature interactions and benefit from additional data to reduce overfitting and improve generalization. In contrast, XGBoost's tree-based ensemble structure, with its inherent robustness to data variability and strong regularization, makes it less sensitive to the size and diversity of the training data. Overall, FNN requires more extensive data (through data aggregation) to fully extract meaningful patterns, while XGBoost maintains stable performance even with limited data, explaining the more modest benefits from data aggregation.

3.2.2 Linear Mixup Data Augmentation

The results for the slider datasets augmented by the two Mixup schemes previously discussed, one with 100% augmentation and one with 200% augmentation, are presented in Figure 3.6 and Figure 3.7. The plots primarily illustrate the impact of Mixup: bars are green if the change is positive (lower MAE, higher R^2) and red if the change is negative (higher MAE, lower R^2). The first scheme creates two artificial data points between two real data points, and the second scheme creates one data point between two real data points. Training remained as before with single-tool and dual-tool training.

Because slider production datasets cluster into three major regions, the virtual metrology function needs to classify each data point correctly into their major regions and detect FAILS (act

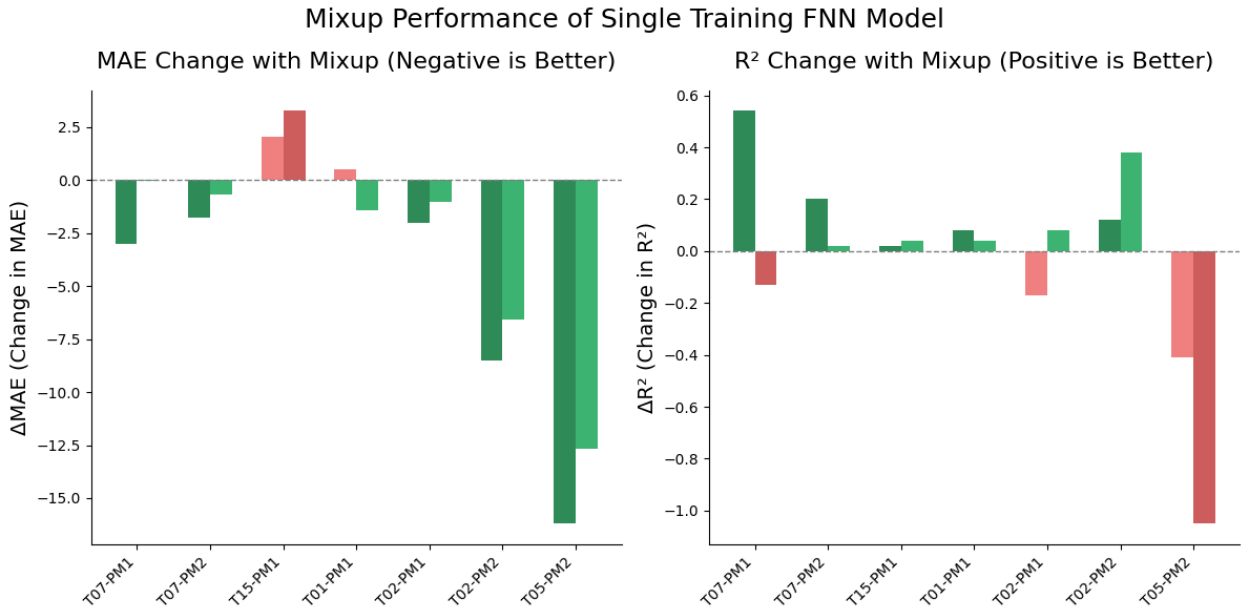

Figure 3.6: MAE and R^2 scores for the FNN model trained with data from a single slider tool using linear Mixup. In general, there is a decrease in MAE and an increase in R^2 scores across most slider tools.

like outlier points in training data). Minimizing the median absolute error (MAE) is therefore crucial to ensure predictions remain within the correct class region. Under the first Mixup scheme for the FNN model, MAE decreased in five tools and increased slightly in two, resulting in an average change of -4.13 (largest decrease: 16.2; largest increase: 2.06). In the second scheme, MAE decreased for all but one tool (T01-PM1), with an average change of -2.73 (largest decrease: 12.7; largest increase: 3.27). The large decrease in MAE especially on T02-PM2 (close to -10) and T05-PM2 (around -15) can significantly improve the multi-class classification accuracy. Although R^2 is less indicative in explaining performance of multi-class classification, it is still useful for assessing the model's ability to capture overall variance. Specifically, the first scheme yielded an average R^2 increase of 0.05 (largest increase: 0.54; largest decrease: 0.41), whereas the second scheme produced an average change of -0.09. The relatively small changes in R^2 suggest that the original model may be adequately explain the variance in the target variable.

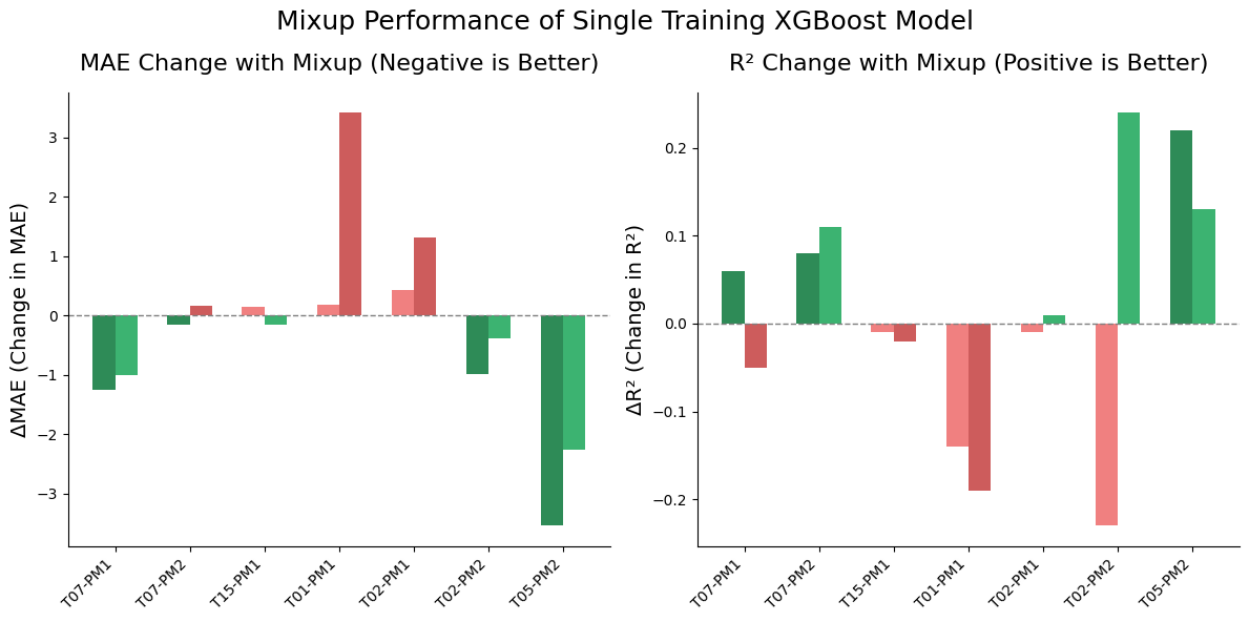

Figure 3.7: MAE and R^2 scores for the XGBoost model trained with data from a single slider tool using linear Mixup. Similar trends of decreased MAE and increased R^2 scores are observed for most tools, except for T05-PM2 where we saw a marked decrease in R^2 scores.

On the other hand, the greater variation in MAE suggests that the model's absolute error is still influenced by the specific conditions of each tool. A higher variation in MAE could indicate that certain tools require additional data to improve prediction stability or that some regions in the feature space are underrepresented. The observed reductions in MAE across most tools confirm that the Mixup augmentation was generally beneficial, but the slight increases in MAE for a few tools highlight areas where additional adjustments, such as targeted data augmentation or feature refinement, may be needed. For the first Mixup scheme with XGBoost, MAE decreased in four tools while increasing slightly in three, yielding an average change of -0.74 (largest decrease (positive): 3.54; largest increase (negative): 0.42). The second Mixup scheme showed a similar pattern, with an average MAE change of 0.15 (largest decrease (negative): 2.27; largest increase (positive): 3.41, this scale of change in MAE won't have a significant impact on the model performance practically. In terms of R^2, results were mixed: the first scheme improved scores for three tools

(average change of -0.004), while the second scheme saw gains in four tools (average change of 0.03). Despite these outcomes, XGBoost benefits less from Mixup compared to FNN, primarily because Mixup generates linearly interpolated data that complements FNNs' continuous, gradient-based learning, helping smooth decision boundaries. By contrast, XGBoost relies on discrete tree splits, where such interpolations offer limited value for split decisions. Furthermore, XGBoost's strong built-in regularization reduces the added benefits of Mixup, while FNNs gain considerable regularization advantages, mitigating overfitting more effectively.

The results for both Mixup schemes on the FNN and XGBoost models using dual training are presented in Figure 3.8 and Figure 3.9. For both figures, each tool has two values for MAE and R^2, the darker hue being the first Mixup scheme and the second value being the second Mixup scheme. A green color indicates a positive change (like a decrease in MAE or increase in R^2) while a red color indicates a negative change (like an increase in MAE or decrease in R^2).

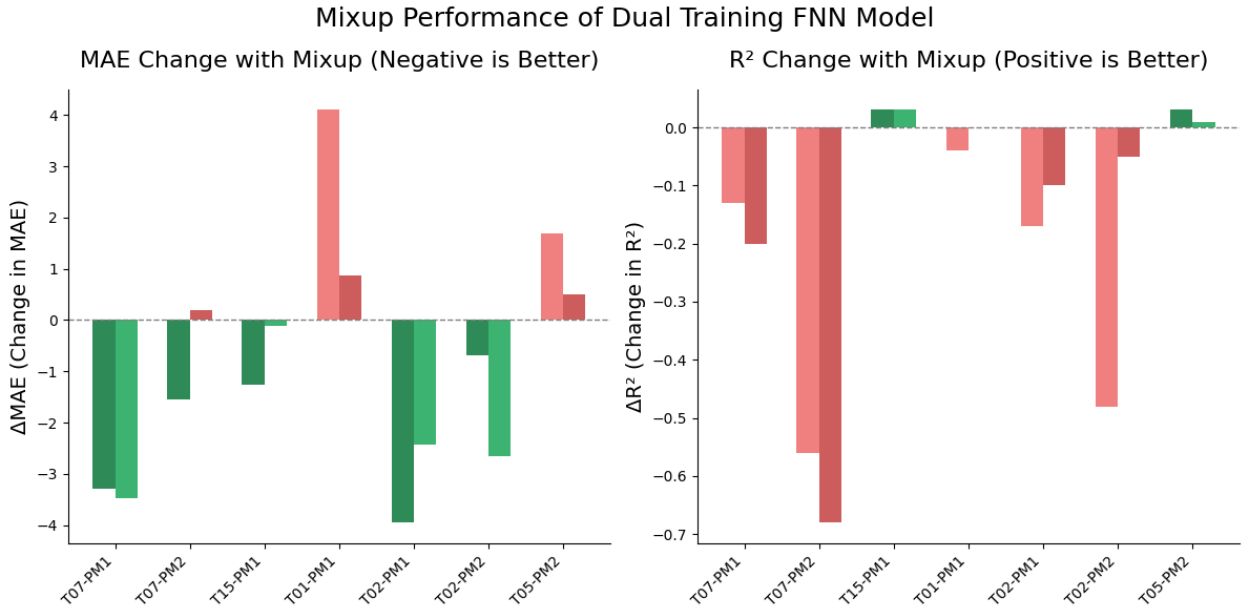

Figure 3.8: MAE and R^2 scores for the FNN model trained with data from two slider tools using linear Mixup. There is a decrease in MAE in 5 tools and an increase in 2 tools for both schemes. R^2 scores decreased in 5 tools for both schemes, with a slight increase in two tools.

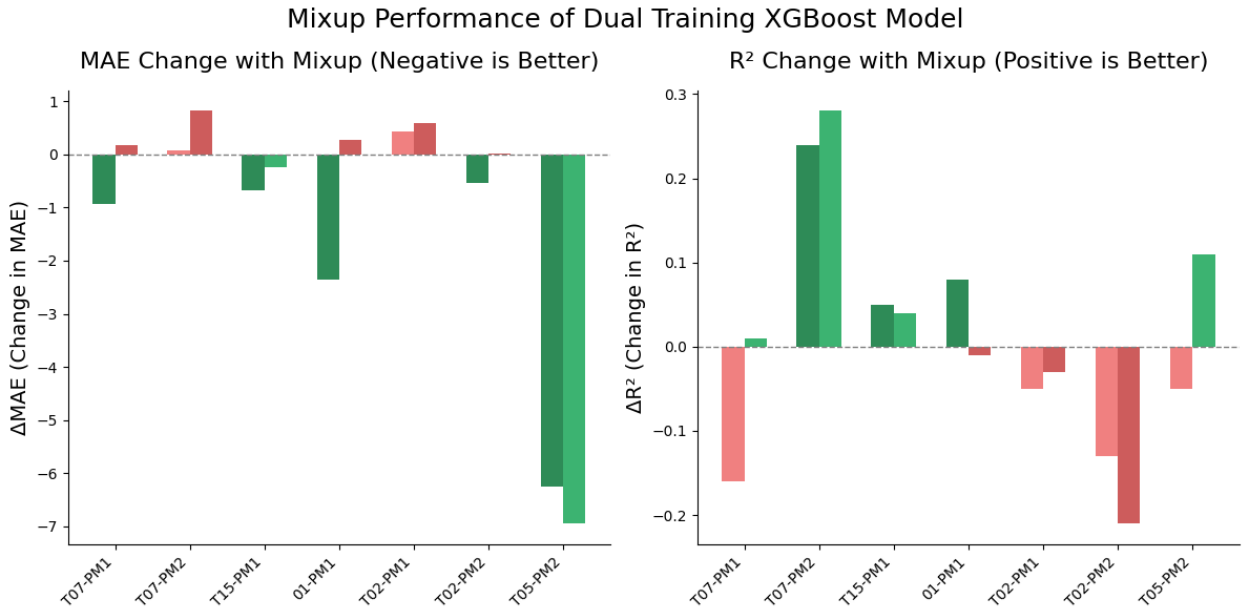

Figure 3.9: MAE and R^2 scores for the XGBoost model trained with data from two slider tools using linear Mixup. MAE generally decreased across most tools, with increases being very slight. R^2 scores saw a mixed pattern of increasing and decreasing scores.

In the first Mixup scheme for the FNN model, MAE decreased for five tools and increased for two, averaging a change of -0.71 (largest decrease: 3.95; largest increase: 4.1). Under the second scheme, MAE decreased in four tools and increased in three, yielding an average change of -1.01 (largest decrease: 3.47; largest increase: 0.87). Although the change in MAE is not large enough to significantly impact real-world model performance, it is still worth noting that Mixup in data aggregation has strong potential for improving model performance. Meanwhile, R^2 declined in five tools and rose slightly in two for both schemes, with average changes of -0.19 and -0.14, respectively, indicating that improvements in error reduction did not consistently translate into higher overall variance capture.

Under the first Mixup scheme with XGBoost, MAE decreased in five tools while rising slightly in two, averaging a change of -1.43 (largest drop: 6.26; largest increase: 0.44). In the second scheme, MAE fell for two tools and rose for five, though some increases were under 1%.

The largest decrease was 6.95, with an average change of –0.76. The decrease of MAE on T05-PM2 is significant enough to improve the model performance, but the changes in other tools are negligible. For R^2, the first scheme raised scores in three tools, resulting in an average change of -0.003, while the second scheme improved four tools with an average increase of 0.03. The largest R^2 gain was 0.28, and the largest decrease was 0.21.

The summary of the overall model performance change results of MAE and R^2 above is tabulated in table 3.1 and table 3.2. The first value in each cell is mean change, while the second value (blue) is median change:

Table 3.1: MAE Change

	FNN		XGB	
Scheme	Single	Dual	Single	Dual
Scheme 1	-4.13/-2	-0.71/-1.26	-0.74/-0.15	-1.43/-0.67
Scheme 2	-2.73/-1	-1.01/-0.12	0.15/-0.16	-0.76/0.18

Table 3.2: R2 Change

	FNN		XGB	
Scheme	Single	Dual	Single	Dual
Scheme 1	0.05/0.08	-0.19/-0.13	-0.004/-0.01	-0.003/-0.05
Scheme 2	-0.09/0.04	-0.14/-0.17	0.03/0.01	0.03/0.01

These results demonstrate the varying impact of linear Mixup data augmentation across different models and training schemes by demonstrating the mean and median value of change in MAE and R^2 after Mixup. While improvements are observed in both metric scores in many cases, the effectiveness of the Mixup technique appears to depend on both the model type and data aggregation status.

In the FNN model, dual training with linear Mixup produced mixed results. For the first scheme, MAE decreased across five tools with an average reduction of 17.0%, while increases

were observed in two tools, averaging 23.6%. Similarly, for the second scheme, MAE decreased by an average of 16.6% in five tools and increased by 4.4% in two tools. In terms of R^2 scores, the results were less favorable, with decreases in five tools and only slight increases in two tools under both schemes. The largest increase of R^2 score in both schemes was 0.03 while the largest decrease was -0.68. These findings suggest that while Mixup can reduce errors, it may not always lead to improved model fit, particularly when the models are trained on data from two tools.

In contrast, the XGBoost model demonstrated more stable performance with Mixup applied under dual training. For the first scheme, MAE decreased across five tools by an average of 12.8%, while the increases, observed in two tools, averaged only 2.6%. The second scheme showed a less favorable balance, with MAE decreasing in two tools (average reduction of 11.4%) and increasing in five tools, though some increases were minor (average increase of 3.8%). For R^2, the results showed mixed trends: under the first scheme, R^2 increased for three tools and decreased for four tools, whereas in the second scheme, four tools saw an increase and three tools experienced a decrease. The largest increase of R^2 score in both schemes was 0.28 while the largest decrease was -0.21.

The differences between the FNN and XGBoost models highlight the role of model architecture in how linear Mixup affects regression performance. FNN models appear to be more sensitive to both improvements and degradations in MAE and R^2, due to their reliance on complex feature relationships that can be influenced by synthetic data interpolation. XGBoost, with its tree-based architecture, shows smaller variations, due to its robust splitting criteria and regularization mechanisms.

According to Table 3.1 and Table 3.2, the impact of linear Mixup on regression performance differs significantly between FNN and XGBoost models. For XGBoost, the mean and median differences in MAE is much smaller compared to FNN for single training, and slightly smaller than FNN for dual training; the R^2 value change is negligible. The small MAE and R^2 changes indicate that Mixup has little effect on its performance. This can be attributed to XGBoost's tree-

based architecture, which relies on robust splitting criteria and inherent regularization mechanisms that make it less sensitive to interpolated synthetic data.

In contrast, the FNN model shows more pronounced effects from Mixup. In single-tool training, Mixup leads to notable improvements in both MAE and R^2, with performance enhancing as more synthetic data points are added (Scheme 1). This could be due to Mixup acting as an effective regularizer, improving generalization by encouraging the model to learn smoother decision boundaries and reducing overfitting to noisy patterns in the original data. However, in dual-tool training, Mixup with a smaller synthetic dataset (Scheme 2) yields better results. This could be because Mixup operates directly at the data level, interpolating between samples from two potentially different data distributions. When the distributions diverge significantly, excessive synthetic data may introduce conflicting patterns, making it harder for the model to generalize effectively. Thus, a smaller Mixup dataset can preserve the distinct characteristics of each tool's data while still providing regularization benefits to achieve lower MAE.

Chapter 4

Conclusion

This paper presents an analysis of the data quality and engineering requirements for machine learning-based virtual metrology for common machine tools in the semiconductor industry. The virtual metrology application studied was the binary PASS/FAIL classification of the product quality at the end of a machine tool step. Two multi-line, multi-machine semiconductor manufacturing operations were studied: five plasma etching tools used in separate wafer production lines and seven milling tools used in separate slider production lines. The study focused on optimum processing of the data by considering the conditions, functions, and data requirements of the machines together and how to leverage commonality.

Feedforward Neural Network (FNN) and XGBoost algorithms were used and compared for both the wafer and slider production machines. For wafer production the algorithms were applied directly for PASS/FAIL classification. For slider production the algorithms were formulated as regression models to predict product thickness that stratified into three clusters for different products. Outlier identification was used to then classify PASS or FAIL. The following data processing/engineering approaches were examined:

- Data aggregation across one or more machines to increase variation and operational coverage.

- Dimensionality reduction techniques were applied to reduce noise and enhance feature extraction.
- A separate scaling approach was implemented during data aggregation to improve model performance by aligning the feature distributions of different datasets and to mitigate discrepancies caused by varying single-feature distributions despite similar feature dependencies.
- Linear Mixup algorithm was applied to augment data and address the scarcity of collected data from slider production tools
- FNN with transfer learning with live model update was used for plasma etching to address process drift. Varying train/test and data length ratios were also tested.

For the wafer production machines, the XGBoost algorithm demonstrated better or comparable performance to FNN in both single-tool and dual-tool training while requiring fewer computational resources and offering greater robustness, making it the better choice for this solution objective. Transfer learning significantly improved model performance across all tested datasets, demonstrating the effectiveness of live updating. Notably, using only 20% of new online data for updates led to substantial performance gains, reducing the need for frequent offline physical measurements and associated costs. Transfer learning also overrode the advantages of data aggregation. In general, given the operational drift with the wafer production tools, transfer learning proved to be a fruitful endeavor.

For the slider tools, XGBoost consistently outperformed or matched FNN in regression tasks in terms of Median Absolute Error (MAE) and R2. Given better performance for both slider and wafer production, XGBoost demonstrated versatility for both classification and regression tasks. Due to its tree-based structure and inherent regularization from ensemble learning, XGBoost was less sensitive to data aggregation compared to the FNNs, though data aggregation still yielded positive effects across all datasets. Mixup strategies demonstrated notable performance improvements

for the FNN models, particularly for single-tool training, due to its regularization effect which promoted better generalization. In contrast, Mixup impact on XGBoost was minimal likely because of XGBoost's tree-based architecture and robust splitting criteria. With respect to dual-tool aggregation, FNN performed better with smaller synthetic datasets, possibly due to distribution differences between datasets that can introduce conflicting patterns when overly mixed. For XGBoost, modest MAE improvements were observed, indicating a resilience to data variations. Overall, the effectiveness of Mixup depends on model architecture and data characteristics.

While no single method universally outperforms others, the combination of Mixup, data aggregation, and transfer learning generally enhanced model performance. When systematically applied as a data engineering procedure, the combination of methods offered significant improvement to this in industrial virtual metrology PASS/FAIL application.

Bibliography

[1] F. Ou, H. Wang, C. Zhang, M. Tom, S. Bom, J. F. Davis, and P. D. Christofides. Industrial data-driven machine learning soft sensing for optimal operation of etching tools. *Digital Chemical Engineering*, 13:100195, 2024.

[2] Casanova. Chip sales rise in 2022, especially to auto, industrial, consumer markets, March 2023.

[3] T. Q. Nguyen, T. Hoang, L. Zhang, O. A. Dobre, and T. Q. Duong. A survey on smart optimisation techniques for 6g-oriented integrated circuits design. *Mobile Networks and Applications*, pages 1–18, 2024.

[4] C. A. Mack. Fifty years of moore's law. *IEEE Transactions on semiconductor manufacturing*, 24(2):202–207, 2011.

[5] W. Mohammad, A. Elomri, and L. Kerbache. The global semiconductor chip shortage: Causes, implications, and potential remedies. *IFAC-PapersOnLine*, 55(10):476–483, 2022.

[6] X. Wu, C. Zhang, and W. Du. An analysis on the crisis of "chips shortage" in automobile industry——based on the double influence of covid-19 and trade friction. In *Journal of Physics: Conference Series*, volume 1971, page 012100. IOP Publishing, 2021.

[7] H. Malkani and P. Korambath. Special issue: Smart manufacturing. *Journal of Advanced Manufacturing Processes*, 2022. Special Issue.

[8] J. Davis, H. Malkani, J. Dyck, P. Korambath, and J. Wise. Chapter 4 - cyberinfrastructure for the democratization of smart manufacturing. In M. Soroush, M. Baldea, and T. F. Edgar, editors, *Smart Manufacturing*, pages 83–116. Elsevier, 2020.

[9] A. Tsanousa, E. Bektsis, C. Kyriakopoulos, A. G. González, U. Leturiondo, I. Gialampoukidis, A. Karakostas, S. Vrochidis, and I. Kompatsiaris. A review of multisensor data fusion solutions in smart manufacturing: Systems and trends. *Sensors*, 22(5):1734, 2022.

[10] J. F. Davis, S. Biller, J. S. Pierre, and S. Jahanmir. Towards resilient manufacturing ecosystems through artificial intelligence. Symposium Report 100-47, National Institute of Standards and Technology (NIST), Gaithersburg, MD, 2022.

[11] S. N. Songkhla and T. Nakamoto. Overview of quartz crystal microbalance behavior analysis and measurement. *Chemosensors*, 9(12):350, 2021.

[12] P. Kadlec, B. Gabrys, and S. Strandt. Data-driven soft sensors in the process industry. *Computers & chemical engineering*, 33(4):795–814, 2009.

[13] G. Wang, Q.-S. Jia, M. Zhou, J. Bi, J. Qiao, and A. Abusorrah. Artificial neural networks for water quality soft-sensing in wastewater treatment: a review. *Artificial Intelligence Review*, 55(1):565–587, 2022.

[14] Y. Jiang, S. Yin, J. Dong, and O. Kaynak. A review on soft sensors for monitoring, control, and optimization of industrial processes. *IEEE Sensors Journal*, 21(11):12868–12881, 2020.

[15] J. Konyha and T. Bányai. Sensor networks for smart manufacturing processes. *Solid State Phenomena*, 261:456–462, 2017.

[16] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[17] S. Coleman, D. Kerr, and Y. Zhang. Image sensing and processing with convolutional neural networks. *Sensors*, 22(10):3612, 2022.

[18] M. Lee, J. Bae, and S. B. Kim. Uncertainty-aware soft sensor using bayesian recurrent neural networks. *Advanced Engineering Informatics*, 50:101434, 2021.

[19] C. Zhang, J. Yella, Y. Huang, X. Qian, S. Petrov, A. Rzhetsky, and S. Bom. Soft sensing transformer: hundreds of sensors are worth a single word. In *Proceedings of IEEE International Conference on Big Data*, pages 1999–2008, 2021.

[20] L. Zhang, Z. Deng, K. Kawaguchi, A. Ghorbani, and J. Zou. How does mixup help with robustness and generalization? *arXiv preprint arXiv:2010.04819*, 2020.

[21] G. Ranganathan. A study to find facts behind preprocessing on deep learning algorithms. *Journal of Innovative Image Processing*, 3(1):66–74, 2021.

[22] C. Albon. *Machine learning with python cookbook: Practical solutions from preprocessing to deep learning*. " O'Reilly Media, Inc.", 2018.

[23] X. Zheng, M. Wang, and J. Ordieres-Meré. Comparison of data preprocessing approaches for applying deep learning to human activity recognition in the context of industry 4.0. *Sensors*, 18(7):2146, 2018.

[24] A. Rehmer and A. Kroll. On the vanishing and exploding gradient problem in gated recurrent units. *IFAC-PapersOnLine*, 53(2):1243–1248, 2020. 21st IFAC World Congress.

[25] J. P. Card, M. Naimo, and W. Ziminsky. Run-to-run process control of a plasma etch process with neural network modelling. *Quality and Reliability Engineering International*, 14:247–260, 1998.

[26] E. A. Elsayed. Overview of reliability testing. *IEEE Transactions on Reliability*, 61(2):282–291, 2012.

[27] B. P. Salmon, W. Kleynhans, C. P. Schwegmann, and J. C. Olivier. Proper comparison among methods using a confusion matrix. In *Proceedings of IEEE International Geoscience and Remote Sensing Symposium*, pages 3057–3060, Milan, Italy, 2015.

[28] L. Gonçalves, A. Subtil, M. R. Oliveira, and P. de Zea Bermudez. ROC curve estimation: An overview. *REVSTAT-Statistical Journal*, 12(1):1–20, 2014.