UNIVERSITY OF CALIFORNIA

Los Angeles

Multiscale Modeling, Computation and Control of Atomic Layer Manufacturing Systems

A dissertation submitted in partial satisfaction of the

requirement for the degree Doctor of Philosophy

in Chemical and Biomolecular Engineering

by

Henrik Wang

2025

ABSTRACT OF THE DISSERTATION

Multiscale Modeling, Computation and Control of Atomic Layer Manufacturing Systems

by

Henrik Wang

Doctor of Philosophy in Chemical and Biomolecular Engineering

University of California, Los Angeles, 2025

Professor Panagiotis D. Christofides, Chair

In modern times, the main driver behind most technological and societal advances is the improvement and further integration of semiconductor computer chips. This results in two types of growing demand: one for higher quality semiconductor devices, and another for more semiconductor chips. Thus, there is a need to optimize and innovate semiconductor fabrication techniques that can produce complex, nanoscale structures and high quality thin films at a fast rate. Atomic layer etching (ALE) and atomic layer deposition (ALD) are two common processes seen in the semiconductor fabrication method, as they help ensure a high product quality. However, there is a lack of research on how process control systems can be used to improve their manufacturing throughput. This is especially apparent as the advent of data-driven and machine-learning models has opened up many novel areas of how these new high-speed models can be integrated into manufacturing processes. To that end, this dissertation first investigates the development and implementation of a first-principles multiscale model for an ALD reactor. Then, various simulations are carried out to determine the feasibility of using data-driven methods to predict key process characteristics such as kinetic activity and reaction completion. The predictor model is then incorporated into a real-time endpoint controller, and its interactions with a run-to-run controller are investigated. The

process predictor is also applied to industrial process data to predict process outcomes and optimize manufacturing efficiency by reducing the usage rate of metrology measurement tools. The models showcased in this dissertation show that data-driven models can offer novel enhancements in manufacturing efficiency from more precise process control to reducing measurement steps. As more semiconductor manufacturing methods are developed and implemented, even more efficient data-driven models can be created, leading to a positive feedback loop of manufacturing efficiency.

The dissertation of Henrik Wang is approved.

Carlos G. Morales-Guio

Dante Simonetti

Xiaochun Li

Panagiotis D. Christofides, Committee Chair

University of California, Los Angeles

2025

Contents

List of Figures

List of Tables

Acknowledgments

This dissertation and work contained within would not be possible without the help and support of too many people to count. Among them, I would like to express my gratitude to my Principal Investigator and advisor, Professor Panagiotis Christofides, who supported and guided me throughout my four years as a Ph.D. student. He has served as a major source of inspiration and support in both research and personal development. I deeply appreciate his encouragement and guidance in helping me overcome difficult academic problems and continuously find new ways to push the boundary of research.

I am grateful to my family and friends, who have supported me throughout this journey. I would like to extend special gratitude towards my parents, Jianghai Wang and Biqiong Yang, my brother, Alexander Wang, and my fiancée, Bridgette Ho for their unconditional support and love.

My academic success would not have been possible without my colleagues. Dr. Sungil Yun and Dr. Matthew Tom served as both mentors and friends when I first started as a Ph.D. student. Professor Gerassimos Orkoulas has always been generous with his advice and time in supporting my various research projects. Feiyang Ou was one of my key collaborators, and I could not have gone nearly as far without him. I would also like to express my appreciation to Julius Suherman, Yifei Wang, and Andrea Lin for their support and inspiration. Lastly, I would like to acknowledge all the members of the Christofides Research Group that have supported me in my journey.

I would like to thank Professor Carlos Morales-Guio, Professor Dante Simonetti, and Professor Xiaochun Li for serving on my doctoral committee.

I would also like to acknowledge Seagate's team, Dr. Chao Zhang and Dr. Sthitie Bom, and Professor Jim Davis for their support and help. Without their contributions, my work would not have been as nearly impactful.

Finally, I gratefully acknowledge the financial support for my research from the National Science Foundation (NSF) and the University of California, Los Angeles, through a Dissertation Year Fellowship. I would not have been able to complete my work without their generous contributions.

This dissertation includes excerpts from the following manuscripts:

- Wang, H., M. Tom, F. Ou, G. Orkoulas and P. D. Christofides, ”Multiscale Computational Fluid Dynamics Modeling of an Area-Selective Atomic Layer Deposition Process Using a Discrete Feed Method,” Dig. Chem. Eng., 10, 100140, 2024.

- Wang, H., F. Ou, J. Suherman, M. Tom, G. Orkoulas and P. D. Christofides, "Data-Driven Machine Learning Predictor Model for Optimal Operation of a Thermal Atomic Layer Etching Reactor," Ind. & Eng. Chem. Res., 63, 19693-19706, 2024.

- Wang, H., F. Ou, J. Suherman, G. Orkoulas and P. D. Christofides, "Integration of On-Line Machine Learning-Based Endpoint Control and Run-to-Run Control for an Atomic Layer Etching Process," Dig. Chem. Eng., 14, 100206, 2025.

- Ou, F., 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," Dig. Chem. Eng., 13, 100195, 2024.

Curriculum Vitae

2021–present	Doctor of Philosophy University of California, Los Angeles
2021–2023	Master of Science University of California, Los Angeles
2013–2017	Bachelor of Science University of California, Los Angeles
2017–2020	Process Engineer Maxim Integrated Portland, Oregon

JOURNAL PUBLICATIONS

1. Ou, F., J. Suherman, C. Zhang, H. Wang, S. Bom, J. F. Davis and P. D. Christofides, "Industrial Multi-Machine Data Aggregation, AI-Ready Data Preparation, and Machine Learning for Virtual Metrology in Semiconductor Wafer and Slider Production," Dig. Chem. Eng., 15, 100242, 2025.

2. H. Wang, F. Ou, J. Suherman, G. Orkoulas and P. D. Christofides, "Integration of On-Line Machine Learning-Based Endpoint Control and Run-to-Run Control for an Atomic Layer Etching Process," Dig. Chem. Eng., 14, 100206, 2025.

3. H. Wang, F. Ou, J. Suherman, M. Tom, G. Orkoulas, P. D. Christofides, “Data-Driven Machine Learning Predictor Model for Optimal Operation of a Thermal Atomic Layer Etching Reactor,” Industrial & Engineering Chemistry Research, 63, 45, 2024.

4. 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,” Dig. Chem. Eng., 13, 100195, 2024.

5. H. Wang, M. Tom, F. Ou, G. Orkoulas and P. D. Christofides, “Integrating Run-to-Run Control with Feedback Control for a Spatial Atomic Layer Etching Reactor,” Chem. Eng. Res. & Des., 203, 1-10, 2024

6. H. Wang, M. Tom, F. Ou, G. Orkoulas and P. D. Christofides, "Multiscale Computational Fluid Dynamics Modeling of an Area-Selective Atomic Layer Deposition Process Using a Discrete Feed Method," Dig. Chem. Eng., 10, 100140, 2024
7. M. Tom, H. Wang, F. Ou, G. Orkoulas and P. D. Christofides, "Machine Learning Modeling and Run-to-Run Control of an Area-Selective Atomic Layer Deposition Spatial Reactor," Coatings, 14, 38, 2024
8. F. Ou, F. Abdullah, H. Wang, M. Tom, G. Orkoulas and P. D. Christofides, "Sparse Identification Modeling and Predictive Control of Wafer Temperature in an Atomic Layer Etching Reactor," Chem. Eng. Res. & Des., 202, 1-11, 2024
9. M. Tom, H. Wang, F. Ou, S. Yun, G. Orkoulas and P. D. Christofides, "Computational Fluid Dynamics Modeling of a Discrete Feed Atomic Layer Deposition Reactor: Application to Reactor Design and Operation," Comp. & Chem. Eng., 178, 108400, 2023
10. S. Yun, H. Wang, M. Tom, F. Ou, G. Orkoulas and P. D. Christofides, "Multiscale CFD Modeling of Spatial Area-Selective Thermal Atomic Layer Deposition: Application to Reactor Design and Operating Condition Calculation," Coatings, 13, 558, 2023
11. M. Tom, S. Yun, H. Wang, F. Ou, G. Orkoulas and P. D. Christofides, "Machine Learning-Based Run-to-Run Control of a Spatial Thermal Atomic Layer Etching Reactor," Comp. & Chem. Eng., 168, 108044, 2022
12. S. Yun, F. Ou, H. Wang, M. Tom, G. Orkoulas and P. D. Christofides, "Atomistic-Mesoscopic Modeling of Area-Selective Thermal Atomic Layer Deposition," Chem. Eng. Res. & Des., 188, 271-286, 2022

Chapter 1

Introduction

1.1 Background

The pace of bleeding edge modern technological advancements is closely correlated to the growth of the semiconductor chip industry. This is especially so with the case of machine learning and AI products, where the improved processing power of modern semiconductor chips has played a key role in their viability [87]. The usefulness of semiconductor logic chips has also disseminated into the ubiquity of everyday life; everything from medical products, entertainment devices, and vehicles depend on the utility provided by semiconductor chips [76].

The advancement of high-power semiconductor chips is mainly driven by Moore's Law, which refers to the trend that the transistor density on semiconductor chips will roughly double every two years [2]. In recent times, this trend has been realized through the development of advanced semiconductor processes such as extreme ultraviolet lithography (EUV) [95], atomic layer etching (ALE), atomic layer deposition (ALD) [22, 34], and area-selective processes [48]. These processes allow for an extreme level of precision in chip fabrication and the creation of complex, 3 Dimensional silicon structures such as FinFETS and GAAs [56].

On top of advancements in the precision and quality of semiconductor manufacturing, many

strides in improving the throughput and manufacturing efficiency have also been made. For example, Seagate has used data-driven and machine learning techniques to reduce the number of measurement steps in their process flow, which allows them to produce more product with a shorter lead time [106].

And yet, these advancements are not enough to satisfy the demand for high-quality semiconductor chips. Computer chips in the automobile and health industries continue to face higher demand than supply availability [66]. And on the side of quality, the advent of Graphical Processing Unit-intensive machine learning artificial intelligence software has created even more drive to further push the processing power of semiconductor chips. Thus, there remains a large space for research and development in increasing both the raw capabilities of semiconductor manufacturing and its overall efficiency.

1.2 Atomic Layer Processes

Compared to other cutting-edge fabrication processes, such as EUV that was only recently incorporated into manufacturing process flows [95], ALD and ALE are relatively mature processes that have been used for many years now [22, 34]. Thus, there is a large body of already-existing research on characterizing and understanding the mechanics of common ALE and ALE processes [82, 102]. This lends itself toward research focused on further optimizing these processes, either for fabricating complex semiconductor devices with tight tolerances or for higher manufacturing efficiency.

The general idea behind atomic layer processes is that they are composed of 2 self-limiting half-reactions. The first half-reaction generally modifies the surface of the substrate in preparation for the second half-reaction, which either etches or deposits the desired material on the substrate to finish the process. These half-reactions are described as self-limiting because the reaction cannot proceed beyond the single mono-atomic layer at the top of the substrate [34]. This causes all of

the half-reactions to slow down as the top layer approaches completion with all kinetic activity halting once the entire layer has finished reacting. Then, by cycling back and forth between the 2 half-reactions, an extremely precise amount of substrate can be etched or deposited as desired.

The bulk of this focuses on the atomic layer etching of aluminum oxide, Al_2O_3, from a silicon oxide substrate, SiO_2, due to the wide applications of this process [102]. In the first half-reaction, shown below, gaseous hydrofluoric acid (HF) fluorinates the Al_2O_3 substrate into AlF_3.

$$\mathrm{Al_2O_3\ (s) + 6HF\ (g) \rightarrow 2AlF_3\ (s) + 3H_2O\ (g)} \qquad \text{(Reaction A)}$$

Then, in the second half-reaction, gaseous TMA etches away the fluorinated ALF_3 surface created in the first half-reaction, releasing a gaseous byproduct of dimethylaluminum fluoride (DMAF) [102].

$$\mathrm{2AlF_3\ (s) + 4Al(CH_3)_3\ (g) \rightarrow 6AlF(CH_3)_2\ (g)} \qquad \text{(Reaction B)}$$

Thus, when a full cycle is completed, the overall equation is:

$$\mathrm{Al_2O_3\ (s) + 6HF\ (g) + 4Al(CH_3)_3\ (g) \rightarrow 3H_2O\ (g) + 6AlF(CH_3)_2\ (g)}$$

where HF and TMA are the 2 main reagents used to etch away the Al_2O_3.

A typical reaction progression of a self-limiting reaction can be seen in Fig. 1.1. As defined, the main benefit of self-limiting reactions are that they slow down as they approach completion. However, this slowing effect causes the reaction rate to be highly nonlinear and difficult to predict. In other words, while overprocessing is not a concern, it is difficult to end reactions at the optimal time due to the nonlinear reaction rates.

Thus, there is a need for powerful control systems that can consistently drive ALE processes to the minimum required process time without underprocessing the wafer.

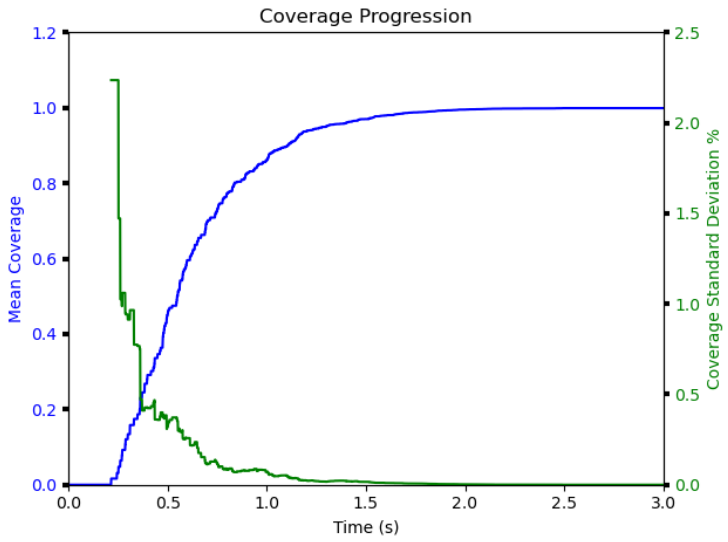


Figure 1.1: Example of the reaction progression through time. Coverage is the fraction of the substrate surface that has finished reacting; a coverage of 1.0 means that the entire surface has finished reacting.

1.3 Multiscale Modeling

While it is critical to continue researching these semiconductor processes, running these reactions is very expensive and often includes the usage of toxic reagents [20, 73]. One way to minimize both the costs and environmental impacts is to employ highly accurate simulations. By employing both first-principles and data-driven methods, accurate simulations of these novel atomic layer processes can be derived and used to offer insight into which process control systems are best suited for them.

The main simulation structure used in this dissertation is that of the multiscale simulation. This method conjoins 2 simulations: a macroscopic Computational Fluid Dynamics (CFD) simulation and a mesoscopic kinetic Monte Carlo (kMC) simulation. Because of the vast timescale differences between them, the pressure fields derived by the CFD simulation can be used as static conditions for the kMC simulation, which calculates the mass source flux caused by the reactions. Since the reaction only occurs on the surface of the wafer, these source terms are used to define the boundary

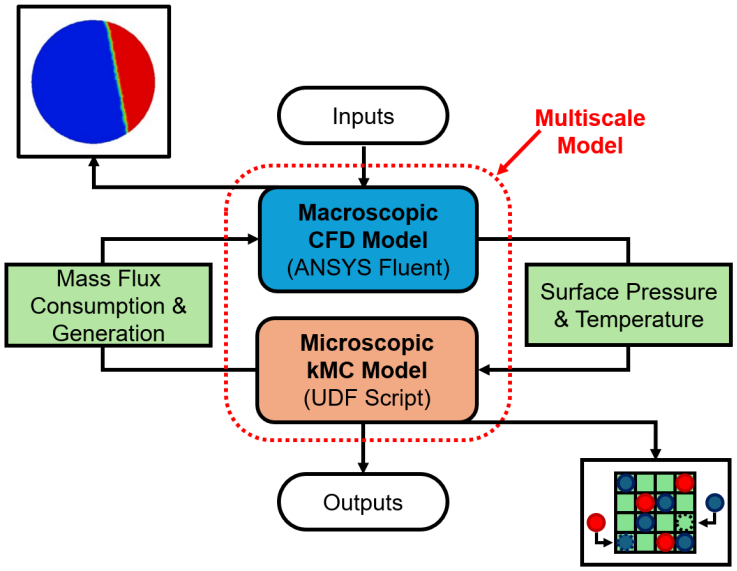


Figure 1.2: Flow diagram of the multiscale model. At each timestep, the CFD model uses the source terms calculated by the kMC model to evaluate the pressure and temperature profiles across the reactor. Then, the kMC model uses the surface pressure and temperature to calculate the source terms for the next timestep.

conditions of the CFD simulation at each timestep. The information flow of the multiscale model is shown in Fig. 1.2.

By running a macroscopic CFD and a mesoscopic kMC simulation together, the multiscale model is able to obtain accurate insights into the mesoscopic reaction kinetics with computational speeds on the order of macroscopic simulations. The accuracy of the simulation is verified by comparing the time it takes to saturate the surface of the wafer with experimental results.

1.4 Data-driven Methods

While the first-principles multiscale model is well-developed and optimized to complete its simulations as quickly and accurately as possible, the simulations are still on the order of hours while the actual process completes within seconds. Thus, it is infeasible to use the multiscale model in any process controller setup, and there is a need for a model that can be computed more

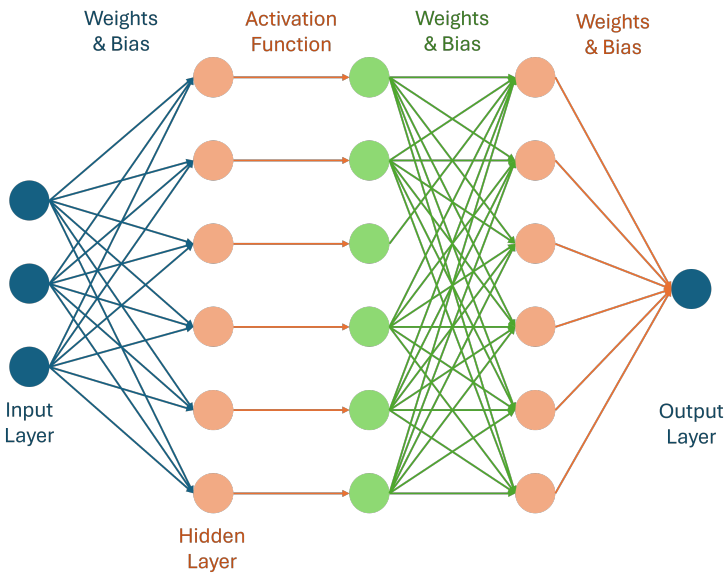


Figure 1.3: The general structure of the Feedforward Neural Network (FNN) is shown here. The hidden layer neurons take a weighted sum from the input layer or the previous hidden layer, which is then activated by nonlinear functions. The output layer takes the weighed sum from the last hidden layer to output the result.

quickly. Data-driven models are ideal at filling this gap because they are both simpler than classic first-principles models and because they can take advantage of the large processing power of a Graphical Processing Unit (GPU) [78]. They can also be applied to industrial process data and extract insights on their reliability. In particular, this dissertation explores the usage of neural networks and transformers to create data-driven models that can quickly predict process outcomes and be used in real-time feedback process controllers.

Neural networks are models that use a certain set of inputs that are multiplied by weights in a hidden layer to predict a certain set of outputs [30]. By training the weights on a large dataset, neural networks can accurately grasp the behavior of nonlinear systems across a large domain of inputs, making them ideal for model-based process controllers.

A general neural network structure is shown in Fig. 1.3. In it, the inputs are entered from the input layer. Then, each neuron in the first hidden layer takes a weighted sum of the inputs, applies a non-linear activation function, and then passes the result to the next layer. The sigmoid function is

often used as the activation function because of its superior performance in classifier models [106]. Subsequent hidden layers take a weighted sum of the previous hidden layer instead of the inputs. The final output layer is essentially a hidden layer with as many neurons as outputs.

In addition to neural networks, a novel deep learning model, the transformer, has emerged in recent years and rapidly gained significant attention in the field of natural language processing [46]. Transformers have also shown equally impressive performances in computer vision tasks such as image classification and object detection [47]. Additionally, one of the most popular and advanced examples of Artificial Intelligence (AI) is in creating large language models such as ChatGPT and BERT [110]. In terms of process data, the transformer structure outperforms feedforward neural networks (FNNs) when it comes to processing time-series data. The former's ability to process long strings of text also allows them to contextualize time-series data within the overall data and capture long-term trends that FNNs struggle with.

The transformer's defining structure is an encoder-decoder architecture that handles sequential data. A multi-head self-attention mechanism [86] is used within each block to compute the relevance of each element in the sequence relative to every other element. This approach enables the model to effectively capture patterns and relations across the entirety of the sequence. It does this with the attention mechanism, which calculates attention scores for each pair of elements and then normalizes them using a softmax function to aggregate their attention values. This process allows the transformer to maintain a comprehensive understanding of the relationships within the data sequence, surpassing the capabilities of traditional FNN networks [16, 86].

1.5 Process Control Methods

The ultimate goal of creating accurate and efficient models of these atomic layer processes is to apply them to advanced process control systems. These systems are a vital aspect of advanced manufacturing methods as they improve process reliability and the manufacturing efficiency by

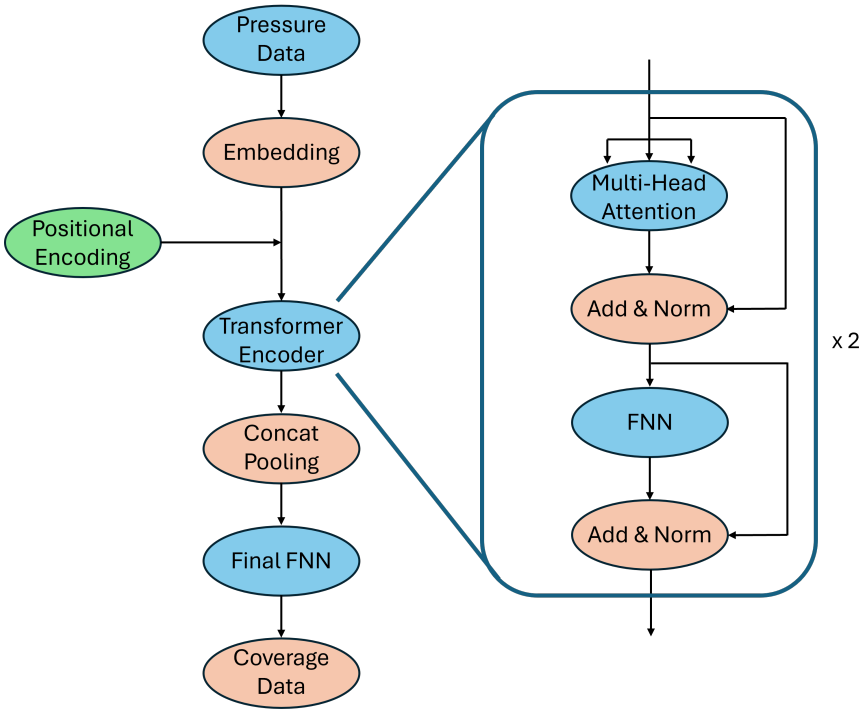


Figure 1.4: The structure of the Transformer model is shown here. Input data is embedded by a dense layer, undergoes positional encoding, and then is fed into 2 identical multi-head encoder blocks. The output of the encoders are combined by concatenation pooling and then fed into the final FNN, which outputs the final reaction coverage.

minimizing the impact of process disturbances on sensitive semiconductor processes.

There are many common disturbances, and they can be sorted into 2 broad categories: drifts and shifts. In drift disturbances, a process variable changes monotonically at a set rate, causing the key process variable to steadily drift away from its target setpoint [54]. For an ALE process, one common drift disturbance is sidewall deposition, where byproducts accumulate on the sides of the reactor walls and disturb the kinetic activity within the reactor. As more product is run, the sidewall deposition becomes thicker and causes a larger disturbance. The other type of disturbance is shift, where a process variable changes in a one-time event, causing the key process variable to suddenly shift away from its target setpoint. This commonly manifests as equipment malfunctions that suddenly and drastically change the key process variable. The main impact of these disturbances is causing key process variables to deviate from their set targets; for an ALE process, this would be the etch per cycle (EPC). If the EPC deviates too far from the target setpoint, the product may have to be thrown away, resulting in wasted resources and time.

For processes like atomic layer etching that have fast reaction dynamics and sensitive responses to disturbances, continuous feedback process control is desired. These online process controllers are able to ensure that the targeted control variable is tightly controlled [80]. However, to enable feedback control, the control variable must be quickly and easily measurable so that the controller can apply control actions in real time. Otherwise, it will be impossible to practically implement the controller. Thus, when a process like ALE has a key process parameter like EPC, solely relying on real-time controllers is not enough.

Another popular control strategy is ex-situ control, which operates on a batch-to-batch basis. These controllers adjust process parameters after a run by using the measured output values from the previous run as feedback, unlike real-time feedback controllers, which make continuous adjustments [54]. While this control method can adjust difficult to measurer parameters such as EPC, one major limitation of R2R control is that adjustments only occur after the completion of an etching cycle. If there is a sudden process shift, then the initially impacted products may have to be

scrapped due to the lack of adjustment for them [29].

1.6 Motivation

This dissertation aims to develop and showcase the possibilities of integrating data-driven models into the existing infrastructure of semiconductor fabrication plants with the atomic layer etching of Al_2O_3 on SiO_2 as an example. First, a multiscale model of the atomic layer process is established. This entails the development and optimization of a macroscopic Computational Fluid Dynamics simulation of the overall Discrete Feed Reactor and a mesoscopic kinetic Monte Carlo simulation of the reaction kinetics on the substrate surface. Then, this multiscale model is used to generate a large amount of process data to train a Transformer model. Specifically, time-series data of the average surface pressure on the substrate is collected and used to predict the kinetic activity on the substrate surface. Finally, this model structure is applied in two ways. First, it is incorporated into a real-time Endpoint Feedback Controller and an ex-situ Run-to-Run Controller that adjust the process time in response to the measured surface pressure of the wafer and the measured etch per cycle. Secondly, it is used with industrial process data to create a process predictor that estimates the likelihood a process will complete successfully given certain process knowledge.

1.7 Dissertation Structure

This dissertation proposes novel uses of data-driven predictors in the semiconductor fabrication industry. In doing so, multiple inquiries into what the best method to acquire the necessary process data to train these models and how to integrate them into existing process control systems are made. The core questions that this dissertation seeks to answer are:

1. How much fidelity is preserved when data-driven models are trained with process data obtained from first-principles models?

2. How can the shortcomings of classic controllers be overcome with data-driven models?

3. How effectively can industrial process data, which is often skewed and noisy, be used to create data-driven models?

4. How robust are process controllers that use data-driven models, and how sensitive to process disturbances are they?

The subsequent sections are organized as follows:

Chapter 2 establishes the multiscale simulation method for an Area-Selective Atomic Layer Deposition reaction in a Discrete Feed Reactor. First, the reaction is modeled on an atomistic level where the thermophysical and structural properties of the reactions are determined. Then, the macroscopic model of the reactor is constructed in Ansys Fluent, a computational fluid dynamics software. Once both the atomistic and the macroscopic models are complete, they are combined into a multiscale model and then subsequently analyzed to determine the optimal reactor geometry that results in the smallest process time.

Chapter 3 studies the development of a process predictor model that uses time-series surface pressure data to predict whether the process is complete or not. First, a multiscale model of an atomic layer etching process is used to generate large amounts of process data. This data is separated into subsets with different reaction kinetic levels to represent different process flows. Transformer models are trained on various combinations of these subsets to explore the effects of data aggregation on improving the robustness of the predictor model and the model's performance on specific datasets.

Chapter 4 explores the interactions between a real-time Endpoint Feedback Controller that uses the data-driven predictor model created in Chapter 3 and a classical Run-to-Run Controller. The Endpoint Controller functions by collecting time-series pressure data and using the predictor model to estimate whether the process is complete or not. If it is, then the process ends; if not, the process continues. This simple logic enables it to control the process time in response to real-

time measurement of a process variable. Then, this data-driven controller is tested in conjunction with an ex-situ Run-to-Run controller that is also trying to minimize the process time to elucidate whether they interact constructively or destructively.

Chapter 5 applies the data-driven methods described in Chapter 3 to industrial data to create a process predictor. Specific care is given to clean the industrial process data to remove incorrect measurements and invalid data points before it is used to train process predictor models. Additionally, process data from different tools are aggregated together to improve the process predictor model for specific tools, and a heuristic to choose which datasets to aggregate to maximize performance on a singular tool is also explored.

Chapter 2

Multiscale Computational Fluid Dynamics Modeling of an Area-Selective Atomic Layer Deposition Process Using a Discrete Feed Method

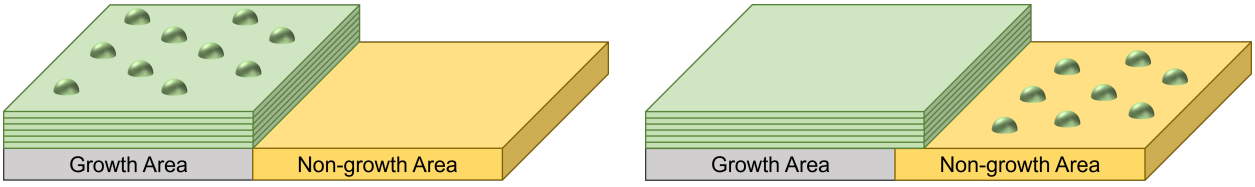
2.1 Introduction

In a growing industrial world comprised of high-performance electronic devices, there exist many challenges in the production of semiconductor devices, which are vital to the innovation of modern-day electronics. The global dependency on semiconductors is appreciable in many industries, particularly so for smart technology, gaming and computing, biomedical technology [8], and communication. However, this technological dependence has also spawned numerous challenges related to consistently manufacturing high-performance semiconductor devices; in part, these challenges stem from the stringent design specifications common for modern semiconductor devices. Additionally, as the rudimentary projections of Moore's Law [53] for the densification of transistor

materials predicted, the miniaturization of transistor length scales has further reduced the capabilities of chip production. Particularly, the continual reduction in transistor size has magnified short-channel effects, which degrade the computing power and power efficiency of transistors [67]. To resolve this issue, complex transistor designs, such as gate-all-around (GAA), have been developed to address various design flaws including power and current losses, thereby optimizing the performance of the transistors [108]. However, transistor fabrication, especially at the nanoscale level, are difficult to implement in large-scale applications [99]. Thus, there is a growing motivation to improve the accuracy and efficiency of the production method of these semiconducting materials. One method to achieve this is to optimize the process by gathering a large amount of data by examining chemical processes and then developing reactor models that enable process scale-up. This chapter will study the development of a three-dimensional reactor with a discrete feed mechanism for an area-selective atomic layer deposition process.

In atomic layer deposition (ALD) processes, precursor reagents are deposited onto the substrate surface to enable thin-layer growth in a bottoms-up fabrication method [107]. Despite ALD being practical in most industrial applications, the process generally introduces alignment issues that are attributed to nonuniform surfaces and growth on non-growth areas, sections of the substrate that do not require deposition, as depicted in Fig. 2.1. Thus, atomic layer etching (ALE) processes were developed to improve the film surface quality in a top-down fabrication approach, but this results in additional processing time, which reduces the overall product throughput. A manufacturing solution, known as area-selective atomic layer deposition (AS-ALD), is an alternative procedure that deposits precursor reagents that only bind to the growth areas of the substrate due to the addition of a chemoselective inhibition step that hinders precursor adsorption on non-growth areas [59]. AS-ALD is desirable in industrial contexts due to its usage of self-aligned structures that facilitate transistor stacking and densification [36]. To ensure the efficacy of AS-ALD and to enable this process in industrial applications, optimal operating conditions must be determined through experimental data, which is time-consuming. Thus, this chapter proposes an

in silico multiscale modeling framework to sufficiently characterize the AS-ALD of an Al_2O_3/SiO_2 (non-growth/growth area) substrate and design a reactor configuration to allow for the scale-up of this process in relevant industrial settings.



(a) (b)

Figure 2.1: Nonideal atomic layer deposition generating (a) nonuniform surface uniformity in the growth area and (b) growth in the non-growth area of the substrate.

The development of multiscale models for thin-layer deposition processes are beneficial to the generation of large datasets, but they require a complex cross-platform programming network that couples various simulations into a single framework [50]. Multiscale models apply a combination of atomistic modeling through *ab initio* quantum mechanics computations to evaluate molecular and kinetic property data, mesoscopic modeling to characterize the stochastic surface kinetics through kinetic Monte Carlo methods, and macroscopic modeling to study the spatiotemporal behavior of fluids through computational fluid dynamics. This type of *in silico* modeling framework is beneficial towards studying the behavior of the AS-ALD process through various time and length scales and towards optimizing reactor configurations with large datasets. This chapter will study the effects of the reactor geometry by examining various discrete feed reactor configurations with the goal of determining the optimal delivery system to produce a high-quality thin film with minimal processing time.

This chapter is organized as follows: Section 2.2 examines the atomistic modeling of structural, electronic, and thermophysical properties and the mesoscopic modeling of surface scale kinetics, Section 2.3 discusses the development of the macroscopic CFD model of an AS-ALD reactor through Ansys Fluent, Section 2.4 elucidates the multiscale modeling methodology used to conjoin the atomistic-mesoscopic and macroscopic simulations, and Section 2.5 analyzes the mul-

tiscale simulation results to determine the optimal reactor geometry that yields minimal process time for achieving full coverage and surface uniformity.

2.2 Atomistic and Mesoscopic Modeling

A vital component to understanding the AS-ALD process lies in the kinetics of the surface reactions. In this chapter, a kinetic Monte Carlo model (kMC) is used to characterize the stochastic nature of surface reactions and determine their dependency on pressure and temperature. This procedure is conducted by first using atomistic modeling techniques via *ab initio* quantum mechanics simulations to derive the reaction rates of possible surface reactions. Then, a kMC algorithm is developed that replicates surface kinetics through a user-defined grid that represents a larger swath of the wafer surface and determines probable reaction pathways for each site on the grid.

The mesoscopic model is one of two integral components in the overall multiscale simulation. Based on the partial pressures and temperature on the surface of the wafer, the extents of reaction in one integration timestep, 0.001 s, are simulated. From this information, the macroscopic model can calculate how much reagent is consumed and how much product is produced, which is accounted for in subsequent timesteps.

2.2.1 Reaction Rate Calculations

The AS-ALD process examined in this chapter comprises three steps: (A) inhibition, (B) precursor adsorption, and (C) oxidation cycle. This chapter studies the AS-ALD of an Al_2O_3/SiO_2 substrate using acetylacetone (Hacac) as a small molecule and gaseous inhibitor for Step A, bis(diethylamino)silane (BDEAS) as a gaseous precursor for Step B, and ozone (O_3) as a gaseous oxidant for Step C. To characterize all three steps of the AS-ALD process, this chapter simplifies the complex reaction mechanisms by concentrating on rate-limiting reaction steps determined through *in silico* modeling works [49, 52, 101] and considering each reaction step as an elementary reaction.

All the reactions involved in the AS-ALD process can be classified into two types: adsorption and nonadsorption. Adsorption reactions can be modeled as bimolecular reactions, and as a result, their reaction rate constants, $k_{ads,s}$, for an adsorbate s, can be calculated through Collision Theory of gases. The aforementioned pressure and temperature-dependent formulation is described as follows:

$$k_{ads,s} = \frac{P_s A_{site} \sigma_s}{Z_s \sqrt{2\pi m_s k_B T}} \tag{2.1}$$

where P_s is the partial pressure of the gaseous reagent s, A_{site} is the surface area of a single active site, σ_s is an experimentally determined sticking coefficient unique to the reagent s, Z_s is the coordination number of the gas s, m_s is the atomic mass of the gaseous reagent s, k_B is the Boltzmann constant, and T is the absolute temperature of the ambient environment.

The reaction rate constants of the nonadsorption reactions, k_{nonad}, are calculated with the temperature-dependent Arrhenius equation, as defined by the following equation:

$$k_{nonad} = \frac{k_B T}{h} \exp\left(-\frac{E_{act}}{RT}\right) \tag{2.2}$$

where h is the Planck constant, E_{act} is the activation energy of the reaction, R is the universal gas constant, and T is the absolute temperature of the reaction. The pre-exponential factor is calculated using Transition-State Theory (TST) by assuming that the ratio of the partition functions for the transition state and the reactants is unity [31]. This assumption was validated with experimentally determined process times for observing full surface coverage by [52]. The activation energy is found by first using *ab initio* quantum mechanics computations to optimize molecular and crystalline structures via Density Functional Theory (DFT). Then, the activation energies between the reactants and products are determined through Nudged Elastic Band (NEB) calculations. The aforementioned computations were conducted through the open-source electronic-structure optimization software Quantum ESPRESSO (QE) in a previous work by [101]. It is notable that all of these k values are the reaction rates for a single active site and have units of s^{-1}.

2.2.2 Surface Kinetics Modeling

The kMC algorithm is a stochastic method that uses a set of randomly generated numbers to simulate the random nature of mesoscopic surface reactions in a spatiotemporal manner [7]. The algorithm used in this chapter is based on the BKL formulation created by Bortz, Kalos, and Lebowitz [4] and has been modified to provide additional insight into the process when the process has larger reaction times relative to the timestep of the overall multiscale simulation.

Algorithm 1 contains the critical steps of the kMC methodology employed in this chapter, which is performed alongside many functions that allow the program to communicate with the macroscopic simulation. Specifically, there are functions that use the macroscopic pressure and temperature data to calculate the reaction rates, which are used as inputs for the kMC algorithm. Additionally, after the algorithm is completed for a given timestep, the new grid data is converted into the change in coverage, which is then used to calculate the generation and consumption source terms for the reactants and products.

There are two major concerns that the modified kMC algorithm must address. The first appears when one of the reactions has a k value magnitudes smaller than that of any other reaction; i.e., the rate-limiting step is substantially slower compared to the remaining reactions. A consequence of this order-of-magnitude difference is that reactions that are more *rate-determining* will contribute more to the overall processing time. For instance, in line 19 of Algorithm 2, which is the algorithm of the BKL kMC method, the size of the time advancement is directly proportional to $1/k$. Thus, when $1/k$ is larger relative to Δt, the large time advancements will generate incomplete data in the form of a step-wise appearance due to how the algorithm will advance major portions of the grid all at once whenever the rate-limiting step is the only available step.

The second area of concern is the memory usage. The BKL method employs a spatiotemporal approach to study the evolution and conversion of active sites that characterize the surface morphology of the substrate. For example, [15] simulated epitaxial growth using a kMC grid to study the surface morphology after each epitaxial cycle but observed computational constraints

Algorithm 1: Modified kMC algorithm implemented in UDF.

Parameters: $P(x, y, t),\ T(x, y, t)$ ▷ *Determined from CFD*

Input: Grid data, $k_{ads,s}(P, T),\ k_{nonad}(T)$ ▷ *Calculated from Parameters*

Output: Grid data, Δt

$\delta t = 0$

while $\delta t < \Delta t$ **do** ▷ *Running algorithm until kMC timestep is as large as CFD timestep*

  Randomly select a site on the grid

  ▷ *Let there be L possible reactions for the selected site and let k_i represent the ith reaction*

  $k_{tot} = \sum_{i=1}^{L} k_i$

  Randomly select $\gamma_1, \gamma_2 \in (0, 1]$

  for j in $1 : L$ **do** ▷ *Going through each possible reaction to randomly select one*

    if $\sum_{i=1}^{j-1} k_i < \gamma_1 k_{tot} \leq \sum_{i=1}^{j+1} k_i$ **then**

      Execute reaction j

      ▷ *Steric hindrance for Step B is not shown here*

  Determine n, the number of active sites from grid data

  $\delta t = \delta t - \ln(\gamma_2)/(n k_{tot})$ ▷ *Advancing kMC timestep*

that limited the grid sizing to an $L \times L$ grid. In a prior work [105], the kMC algorithm was implemented through an external Python program and conjoined with the macroscopic computational fluid dynamics simulation in Ansys Fluent through a Linux Bash script. For this chapter, the kMC algorithm was directly implemented in Fluent through custom user-defined functions (UDFs) written in the C programming language [1]. While this provides a major increase in computational performance, this method also comes with more restrictions for the new code. The main restriction when implementing UDFs in Fluent is the memory storage, where a maximum of 500 variables can be safely stored for each integration timestep [1]. However, each kMC grid is defined by a 300×300 lattice in this chapter for a total of 90,000 sites. Thus, the data of each kMC grid must be stored differently so that it can be represented by less than 500 variables.

To rectify the two issues presented, the kMC algorithm used in this chapter was modified to evaluate the time required for a single active site to react, whereas the BKL algorithm evaluates how the entire grid progresses in a given timeframe [4]. This difference is implemented in the form of three distinctive changes, which are summarized as follows:

1. The arrangement of occupied sites is neglected by employing a Markov chain in which only one site on the entire grid advances with each step as conducted by [37].

2. The time advancement computation includes the number of unoccupied sites on the grid, where a reduction in unoccupied sites increases the time progression as employed by [25] and [41].

3. Grid data is stored as a single integer variable that counts the number of each species, rather than as an array.

The first two modifications resolve the first concern regarding small k values, and the last modification resolves the second concern regarding memory storage. These adjustments to the BKL formulation are further examined and verified in Section 2.2.2.

Algorithm 2: BKL kMC algorithm implemented in Python.

Parameters: $P(x, y, t),\ T(x, y, t)$ ▷ *Determined from CFD*

Input: Grid data, $k_{ads,s}(P, T),\ k_{nonad}(T)$ ▷ *Calculated from Parameters*

Output: Grid data, Δt

▷ *Let there be X grid rows and Y grid columns*

▷ *Let there be R reactions in the process, and let k_i represent the ith reaction*

$\delta t = 0$

while $\delta t < \Delta t$ **do** ▷ *Running algorithm until kMC timestep is as large as CFD timestep*

  for each species **do**

    if number of species in grid = 0 **then**

      Set the appropriate k_i value(s) to 0 ▷ *Removing impossible reactions*

  $k_{tot} = \sum_{i=1}^{R} k_i$

  for j in $1 : X$ **do**

    for k in $1 : Y$ **do**

      ▷ *Randomly determining if a reaction occurs for each site on the $X \times Y$ grid*

      Randomly select $\gamma_1 \in (0, 1]$

      for r in $1 : R$ **do**

        if $\sum_{i=1}^{r-1} k_i < \gamma_1 k_{tot} \leq \sum_{i=1}^{r} k_i$ **then**

          if reaction r is possible **then**

            Execute reaction r

            ▷ *Steric hindrance for Step B is not shown here.*

  Randomly select $\gamma_2 \in (0, 1]$

  $\delta t = \delta t - \ln(\gamma_2)/k_{tot}$ ▷ *Advancing kMC timestep*

Derivation of the Modified kMC Algorithm

To properly make the necessary modifications to the BKL kMC algorithm, it is first important to understand what the kinetic rate represents. Intuitively, the parameter reflects the number of reactions for a given reaction that occurs every second at an active site, which is a region on the substrate surface that is able to undergo a chemical reaction. In other words, the kinetic rate is a measure of the probability that such an event is happening. Additionally, these events can be assumed to follow a Poisson distribution, to react independently of other sites, and to react independently of other possible reactions [9]. As such, probability theory allows the reaction rate to be decomposed into two independent events [24], as shown in Eq. (2.3). First, there is the probability that the site is in a state where the reaction can proceed, $\mathcal{P}(possible)$. Second, there is the probability that the reaction actually proceeds, $\mathcal{P}(proceed)$. This decomposition is expressed as follows:

$$k_{rxn} = \mathcal{P}(reaction)$$

$$k_{rxn} = \mathcal{P}(possible) \cdot \mathcal{P}(proceed) \tag{2.3}$$

where k_{rxn} is the reaction rate of a given reaction for a single active site as calculated in Section 2.2.1 and $\mathcal{P}(reaction)$ is the probability of that reaction taking place in one second.

A similar expression for the reaction rate of the entire grid can be derived by using the number of sites rather than the probability of a site being able to undergo the desired reaction. This expansion is also based on the assumption that these two events are mutually exclusive, yielding:

$$k_{grid} = \mathcal{N}(possible) \cdot \mathcal{P}(proceed) \tag{2.4}$$

where k_{grid} is the average reaction rate for a given kMC grid and $\mathcal{N}(possible)$ is the number of sites that can undergo the desired reaction.

Because of the assumption that each site on the grid is independent of the others, $\mathcal{P}(possible)$

can be related to $\mathcal{N}(possible)$ as follows:

$$\mathcal{N}(possible) = n \cdot \mathcal{P}(possible) \tag{2.5}$$

where n is the number of active sites on the grid. With this relationship, k_{grid} can be related to k_{rxn} as follows:

$$k_{grid} = \mathcal{N}(possible) \cdot \mathcal{P}(proceed)$$

$$k_{grid} = n \cdot \mathcal{P}(possible) \cdot \mathcal{P}(proceed)$$

$$k_{grid} = n \cdot k_{rxn} \tag{2.6}$$

where k_{grid} is the average reaction rate of a given reaction for the entire grid, k_{rxn} is the reaction rate for a single active site, and n is the number of actives sites on the grid.

When there are multiple possible reactions, it is necessary to determine k_{tot}, which is the probability of an unspecified reaction occurring each second. Because it is assumed that the probability of each k is independent of other reactions, the probability that one of two reactions will take place can be found through the following calculation:

$$\mathcal{P}(k_i \cup k_j) = \mathcal{P}(k_i) + \mathcal{P}(k_j) + \mathcal{P}(k_i \cap k_j)$$

where $\mathcal{P}(k_i \cup k_j)$ is the probability that either reaction i or reaction j will take place, $\mathcal{P}(k_i)$ is the probability that reaction i will occur, $\mathcal{P}(k_j)$ is the probability that reaction j will occur, and $\mathcal{P}(k_i \cap k_j)$ is the probability that both reaction i and reaction j will occur. This equation can be simplified by noting that $\mathcal{P}_i = k_i$ as shown in Eq. (2.3) and that $\mathcal{P}(k_i \cap k_j) = 0$ because the two reactions are mutually exclusive, which yields:

$$\mathcal{P}(k_i \cup k_j) = k_i + k_j$$

If there are R total possible reactions, these additional reactions can be summed into $\mathcal{P}(k_i \cup k_j)$ to obtain the probability that an unspecified reaction occurs as all of these reactions are independent and mutually exclusive. The expression for this possibility is:

$$\mathcal{P}(k_{tot}) = k_{tot} = \sum_{z=1}^{R} k_z \tag{2.7}$$

where k_{tot} is the possibility of an unspecified reaction occurring and k_z represents the possibility for a specific reaction to occur. Note that, when evaluating an active site in isolation, $k_z = k_{rxn}$; similarly, when evaluating an active site in the context of the entire grid, $k_z = k_{grid}$. Thus, to evaluate the time progression for a single active site at a time, $k_{tot,grid}$ can be represented as follows:

$$\begin{aligned} k_{tot,grid} &= \sum_{z=1}^{R} k_{grid} \\ &= \sum_{z=1}^{R} n \cdot k_{rxn} \\ &= n \cdot \sum_{z=1}^{R} k_{rxn} \\ &= n \cdot k_{tot,rxn} \end{aligned}$$

where $k_{tot,grid}$ is the possibility that an unspecified reaction will occur anywhere on the kMC grid, $k_{tot,rxn}$ is the possibility that an unspecified reaction will occur at a singular active site, and n is the number of active sites on the kMC grid. This formula is employed in line 12 of Algorithm 1.

The other modification made to the kMC method presented in [101] is the reduction of memory usage as necessitated by the restrictions of Ansys Fluent. The total data stored in between each timestep must be reduced from 90,000 integers for a 300 $\times$ 300 grid to less than 500 integers. This was done by taking advantage of the fact that the kMC algorithm does not use any positional data; i.e., the simulation is not concerned about the location of the site in the grid, but rather only the state of the site for each timestep. Thus, instead of using an array with 90,000 entries, the amount

of each intermediate species was counted and saved in its own variable. For example, in Step C of the AS-ALD cycle, there are 3 species: V4, V6, and V8. The old method would have 90,000 entries, each one representing a site and tracking whether it is in state V4, V6, or V8. The new method has 3 variables, which are defined as *buckets* that represent the number of V4 sites, the number of V6 sites, and the number of V8 sites. By discarding the unnecessary positional data of the sites, this bucket method is able to store the relevant information of all the kMC grids in 50 variables.

However, the reaction mechanism for Step B is more complicated than the other two reactions, as steric hindrance plays an essential role in the kinetics of that process. The specific details of the effects of steric hindrance on the BKL formulation can be found in [101], but a summary is as follows. Each site has two adjacent neighbors that apply two conditions on the surface reaction mechanism. The first extra condition is that a site is restricted from certain reactions if a bulky molecule has adsorbed to either neighboring site. Physically, these molecules hinder the primary site from reacting. The second condition is attributed to the final surface reaction, which requires 2 adjacent sites to bond and for both sites to reach the final state. Thus, it is possible for situations to arise where it is impossible for a site to fully react if both of its neighbors have reached completion by bonding with other sites. As a result, these sites must be deactivated so that the kMC algorithm can reach completion.

To implement the first condition described above, the modified kMC algorithm creates buckets that represent the number of adjacent sites that are blocking it. Because each site has two sterically relevant neighboring sites, there are 3 block status buckets: unblocked, one-block, and two-block. While both the one-block and two-block status represent the target site being unable to undergo certain surface reactions, distinguishing between the two allows us to preserve more positional data about the grid. Now, during the kMC algorithm, whenever the algorithm needs to determine whether the site it randomly selected is blocked, it will do so by randomly selecting a block status.

The second condition is actually an extra step that takes place after each kMC event. To prop-

erly deactivate sites, the following procedure is taken after each iteration of Algorithm 1. By

Algorithm 3: Step B steric hindrance locking algorithm.

Variables:

V4: Completely reacted site

IC: Site that will never reach completion

LK: 2 adjacent V4s that cannot trap an unfinished site

▷ *Let there be N total sites*

▷ *Let S_i represent the species of site i*

for j in $1 : N$ **do** ▷ *Going through each site on the grid*

    Randomly select a species S_j

    ▷ *Let $S_{adj1},\ S_{adj2}$ be two randomly selected sites that represent the sites adjacent to S_j*

    if $S_j \neq$ V4 AND S_{adj1} = V4 AND S_{adj2} = V4 **then**

        $S_j \longrightarrow$ IC

    else if S_j = V4 AND (S_{adj1} = V4 OR S_{adj2} = V4) **then**

        $S_j \longrightarrow$ LK

        $V4 \longrightarrow$ LK ▷ *This represents the adjacent V4 turning into LK*

running Algorithm 3, the number of trapped sites that are unable to reach the final V4 state can be accurately represented even after discarding all positional data. After implementing both modifications discussed in this section, the kMC model is able to simulate the surface reactions with greater resolution and obtain high quality results for all the reactions in the AS-ALD process.

Verification of the Modified kMC Algorithm

To verify that the results of the modified kMC algorithm are accurate and valid, comparisons between Algorithms 1 and 2 were examined for both cases with slow reactions (Step B) and cases without slow reactions (Steps A and C).

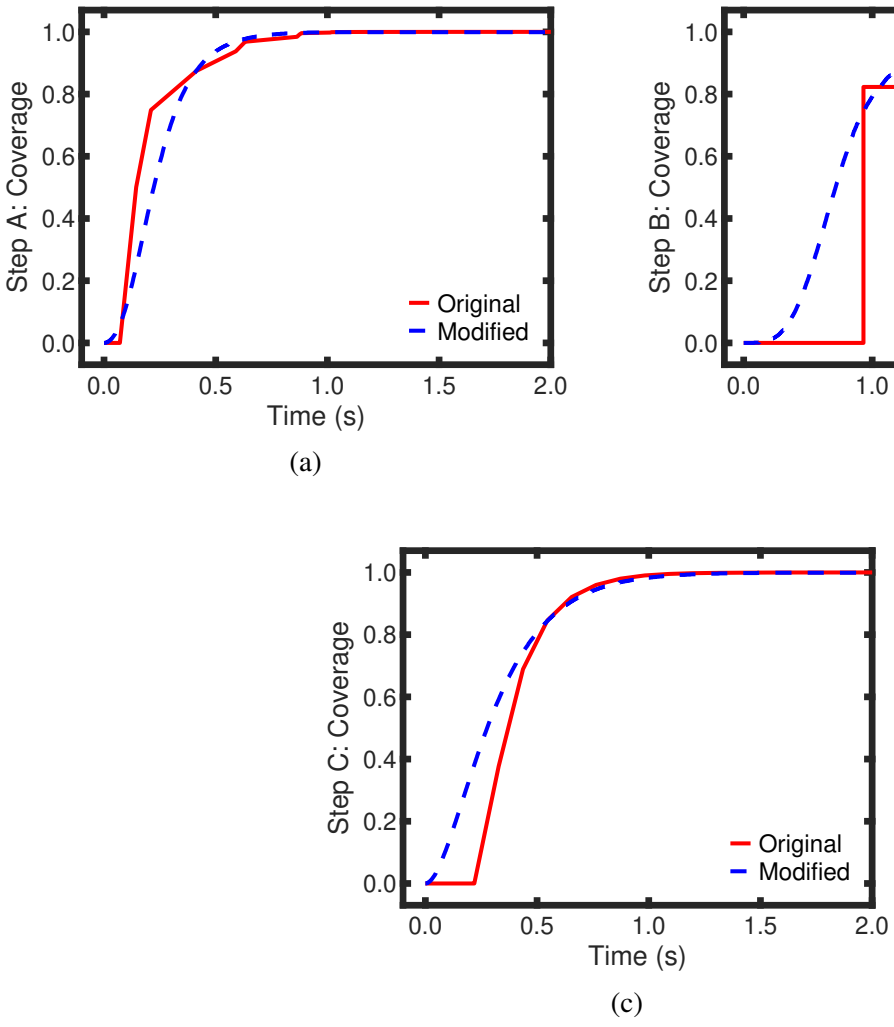


Figure 2.2: Comparison of the original and modified kMC algorithms for Steps (a) A, (b) B, and (c) C. The original kMC processing times to reach full coverage for Steps A, B, and C are 1.018, 2.796, and 1.418 *s*, respectively. The modified kMC processing times to reach full coverage for Steps A, B, and C are 0.969, 2.793, and 1.487 *s*, respectively.

2.3 Computational Fluid Dynamics Modeling

Computational fluid dynamics (CFD) simulations describe the macroscopic behavior of fluids in larger time and length scales, which enables the scale-up of processes. The integration of CFD is applicable to characterizing the spatiotemporal flow of reagents on the substrate surface, which experiences surface reactions that consume the reagents and generate byproducts. The develop-

ment of a CFD model requires the construction of a computer-aided design (CAD) model for a three-dimensional (3D) discrete feed reactor system, the performing of a meshing procedure on the CAD model, and the creation of the CFD simulation of the AS-ALD process in the discrete feed reactor model.

2.3.1 Reactor Design

An abundance of reactor models have been investigated to discuss the uniformity of reagent coverage on the substrate surfaces and to improve the productivity of the process. For example, [103] proposed a cross-flow reactor to control the behavior of flow in the azimuthal direction of the substrate for an atomic layer etching process. Additionally, [17] and [18] suggested using showerhead reactors to improve the uniformity of fluid flow in the radial direction. By considering the challenges attributed to low product throughput, [105] proposed spatial reactor configurations where the reagent is delivered perpendicular to the substrate in a continuous feeding mechanism for atomic layer etching and area-selective atomic layer deposition processes, respectively. While the aforementioned reactor models have effectively yielded valuable results in improving product quality and yield, this chapter characterizes the impact of steric collisions generated from bulky molecular species including Hacac and BDEAS, which introduces challenges associated with surface uniformity. Thus, there is motivation to develop a reactor that minimizes steric hindrance induced by screening effects.

This chapter adopts a previously designed discrete feed reactor [83] inspired by the work of [45] through Ansys DesignModeler, which delivers reagent perpendicularly to the substrate surface in discrete pulses through an injection plate. The employment of discrete feeding with cut-in purging allows the byproduct species that inhibit adsorption of Hacac and BDEAS on the substrate surface to be regularly removed. The discrete feed reactor, illustrated in Fig. 2.3, situates a showerhead divider that is below and parallel to the injection plate to facilitate the transport of reagents in the radial directions of the substrate, thereby maximizing the exposure of the substrate to the reagent

in minimal pulse times. The gap distance between the injection to the showerhead plate is 3 *mm*, and the gap distance between the showerhead plate to the 200-*mm* diameter substrate surface is 5 *mm*. These gap distances are necessary to minimize the volume required to maintain laminar flow behavior [38]. A summary of the reactor dimensions are summarized in Table 2.1.

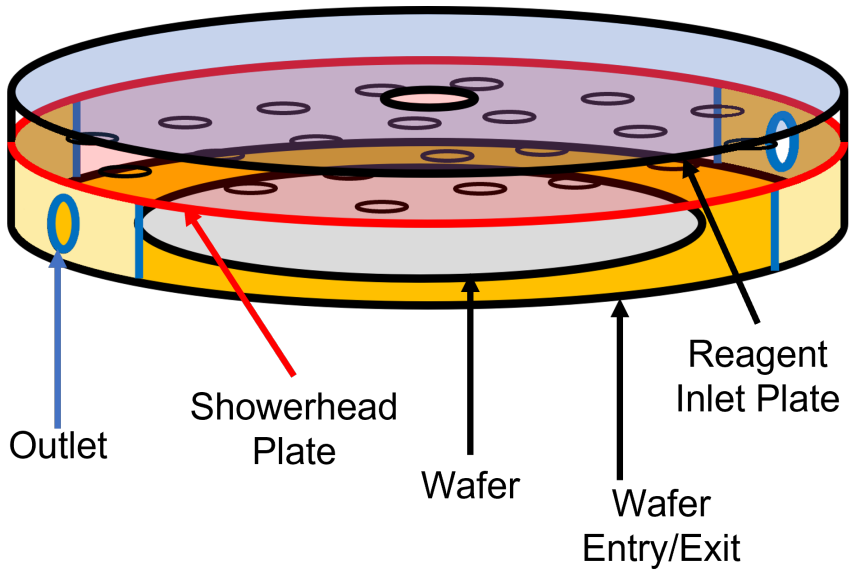


Figure 2.3: Schematic of the discrete feed reactor model for the AS-ALD reaction.

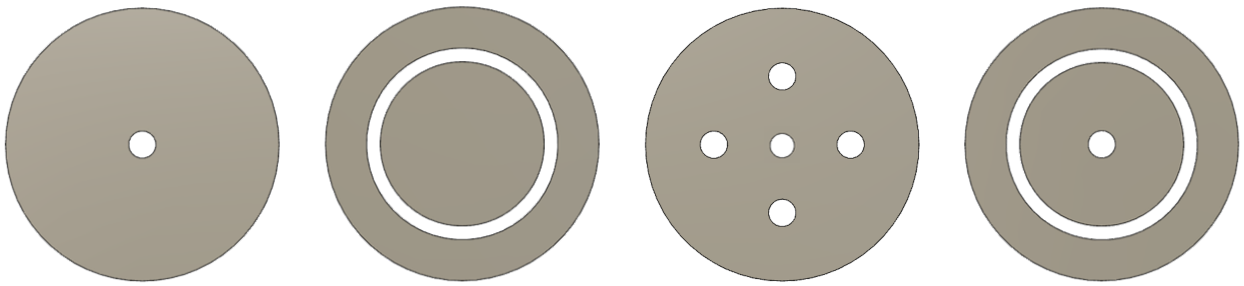
The injection plate has a substantial impact on the mass transport of reagents in the radial direction. Thus, various injection plate geometries, which are illustrated in Fig. 2.4, were previously proposed by [83] to observe their impact on the fluid dynamics on the substrate surface. Results from the aforementioned work provided valuable information about the role of characteristic lengths on the rate of mass transfer in the radial direction. This chapter extends prior macroscopic modeling work for each reactor injection plate geometries by studying their effect on the spatiotemporal coverage and process time required to reach complete surface coverage.

2.3.2 Meshing

Following the construction of the reactor model, a discretization process is conducted to produce conformal meshes that balance computational efficiency and accuracy when performing the

Table 2.1: Dimensions for the reactor configurations.

Reactor Dimension	Value
Plate Diameter	290 *mm*
Ring Inlet Outer Diameter	170 *mm*
Ring Inlet Inner Diameter	130 *mm*
Round Inlet Diameter	20 *mm*
Round Outlet Diameter	4 *mm*
Showerhead Diameter	250 *mm*
Showerhead Pores Diameter	10 *mm*
Showerhead Thickness	0.5 *mm*
Showerhead-Wafer Gap Distance	5 *mm*
Inlet-Showerhead Gap Distance	3 *mm*
Wall Sector Angle	40°



(a) (b) (c) (d)

Figure 2.4: Various feed distributor geometries for (a) Single, (b) Ring, (c) Multi, and (d) Combined reactor configurations.

finite element method. Meshes for each reactor model are produced from "Meshing Mode," a feature of the multiphysics software, Ansys Fluent, in a prior work [83]. The aforementioned meshes were generated by optimizing mesh quality parameters based on the tetrahedral geometries of the discretized cells, which include the orthogonality, aspect ratio, and skewness [1]. To maximize each reactor configuration mesh, optional remeshing tools were then applied to the irregular sur-

face and volume cells. In additional to maintaining balanced mesh quality, this chapter aims to minimize the number of cells required to produce the 3-D meshes to reduce the complexity of the computational fluid dynamics simulation, where each reactor configuration comprises 1.1 to 1.2 million cells.

2.3.3 Computational Fluid Dynamics Simulation Framework

The macroscopic CFD simulation is constructed by defining boundary, operating, and solver conditions that are specific to the AS-ALD process. This simulation will employ a strategy for solving the mass, momentum, and energy transport equations, which are described as follows:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \overrightarrow{v}) = S_m \tag{2.8}$$

$$\frac{\partial}{\partial t}(\rho \overrightarrow{v}) + \nabla \cdot (\rho \overrightarrow{v}\,\overrightarrow{v}) = -\nabla P + \nabla \cdot (\overline{\overline{\tau}}) + \rho \overrightarrow{g} + \overrightarrow{F} \tag{2.9}$$

$$\frac{\partial}{\partial t}(\rho E) + \nabla \left(\overrightarrow{v}\,(\rho E + P)\right) = -\nabla \left(\Sigma h_j \overrightarrow{J}_j\right) + S_h \tag{2.10}$$

where the mass transport equation in Eq. (2.8) is related to the gas-phase species flux, which is represented by the product of the gas-phase species density, ρ, and the velocity of the species, $\overrightarrow{v}$, and is related to the species source generation and consumption flux rate, S_m. The momentum transport equation in Eq. (2.9) relates the rate of momentum per unit volume to the convection, pressure, viscous, and gravitational forces where P is the operating pressure of the reactor, $\overline{\overline{\tau}}$ is the normal two-rank stress tensor, $\overrightarrow{g}$ is the gravitational acceleration constant, and $\overrightarrow{F}$ is the force acting on the system. The energy transport equation defined in Eq. (2.10) describes the relation of the accumulated rate of system energy, E, with the convective, conductive, and energy source generation or consumption, S_h, rates, where h_j and $\overrightarrow{J}_j$ is the sensible enthalpy and mass diffusion flux, respectively, of the gas species j.

Ansys Fluent contains multiple fluid dynamics models that can be used to describe the behav-

ior of the fluid flow. Due to the small reactor sizes and observance of laminar behavior from prior research [83], a laminar fluid model is defined in the simulation. The mass transport is simulated by specifying gas-phase reagent and byproduct species that are present in the Ansys ChemKin database and thermophysical property data generated from experimental works and *ab initio* quantum mechanics calculations discussed in Section 2.2. The source generation and consumption flux rate terms are evaluated through the kMC simulation and defined as boundary conditions on the wafer surface through user-defined functions (UDFs). Additionally, this simulation considered the role of ozone decomposition within the reactor and surface of the wafer. The reactor is also operated under isothermal and isobaric conditions by assuming that a temperature control system is used to maintain the temperature on the wafer surface and that a vacuum pump is effectively applied to regulate the pressure within the reactor chamber.

A pressure-based coupled solver method is integrated into this chapter to simultaneously solve the momentum and pressure-based continuity equations in a parallelized algorithm to reduce computation time at a cost of increased memory requirement. To circumvent this issue, CPU-based (central processing unit) nodes were integrated into this chapter comprising 48 and 36 cores with 512 *GB* and 384 *GB* of dynamic random-access memory (DRAM), respectively, and executed through text-user interface (TUI) commands to minimize graphical power. Additionally, a fixed timestep method of step size 0.001 *s* is defined, which is within the Courant number threshold recommended by the default settings for the program. Lastly, under-relaxation factors of 0.5 were assigned to all gas-phase species involved in the mass transport calculations to minimize the potential for divergent or oscillatory residual responses that could potentially be generated from the source flux rate terms evaluated from the kMC simulation.

Table 2.2: Operating conditions for each reactor geometry.

Reactor	Temperature (K)	Pressure (Pa)	Mole Fraction			Mass Flow Rate (kg/s)
			Hacac	BDEAS	Ozone	
Single	573	300	0.50	0.50	0.20	2.00×10^{-5}
Ring	573	300	0.50	0.50	0.20	2.00×10^{-5}
Multi	573	300	0.50	0.50	0.20	Each Inlet: 4.00×10^{-6}
Combined	573	300	0.50	0.50	0.20	Single: 1.00×10^{-5} Ring: 1.00×10^{-5}

2.4 Multiscale Modeling

The efficacy and impact that simulations can have naturally depends on their accuracy and precision. Generally speaking, the accuracy of a simulation can always be improved by increasing the computational costs; for example, lowering the integration timestep when numerically solving a differential equation will improve the accuracy of the final answer while increasing the number of calculations that must be made to reach that final answer. Thus, one of the driving motivations for this chapter is finding an optimal balance between the accuracy and the computational cost of the simulation.

One commonly used method to improve simulation accuracy without an expensive computational cost is multiscale simulation [94]. This method comprises two simulations that run concurrently and interact with each other: a mesoscopic kMC model that simulates the surface kinetics of the wafer as a function of the pressure and temperature, and a macroscopic CFD model that simulates how the pressure fields within a reactor evolve with time. These two interacting models improve the overall accuracy of the simulation because their domains are intrinsically linked. The surface reactions on the wafer generate and consume products and reagents, which affects the overall pressure fields in the reactor, which affects the reaction rate on the wafer surface. Thus, to improve simulation accuracy, the two models are integrated together in a multiscale framework as

shown in Fig. 2.5.

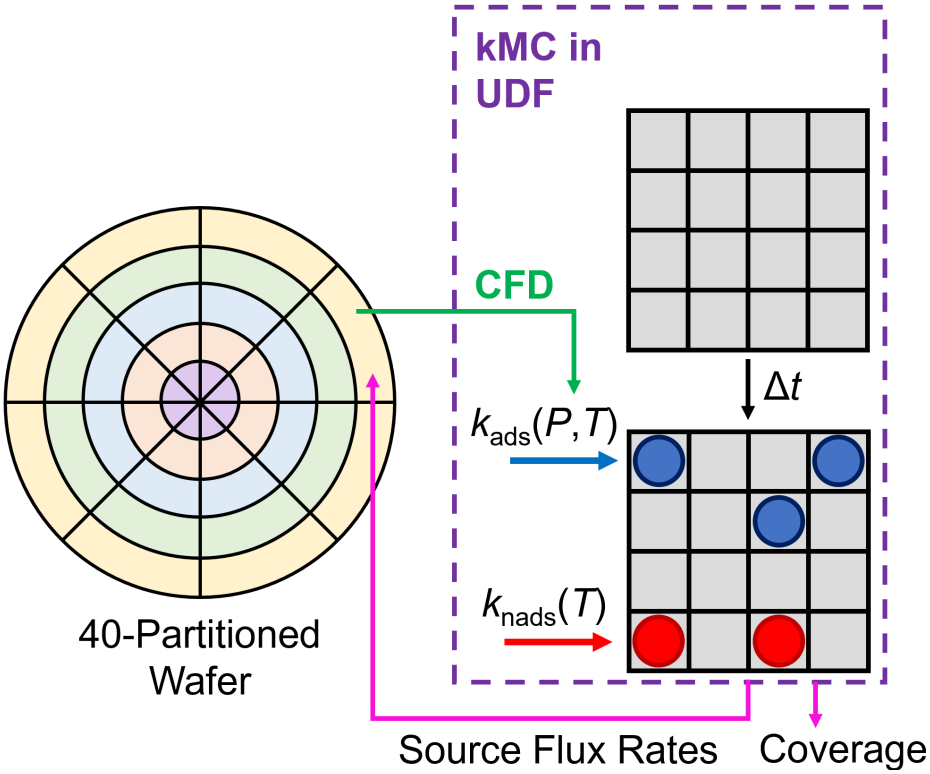


Figure 2.5: Illustration of the multiscale CFD modeling framework. The wafer is partitioned into 40 sections in the CFD simulation to produce a collection of 40 surface pressure and temperature datasets that are used to calculate the reaction rate constants in user-defined functions (UDFs). The kMC simulation, which is performed in the UDF calculates the surface coverage and source flux rate terms that are transmitted to the CFD simulation.

At each integration timestep, $\Delta t = 0.001\ s$, the CFD model takes the generation and consumption terms calculated by the kMC model in the previous timestep into account when calculating the pressure fields in the reaction. Then, the kMC model receives information about the species pressure and temperature at the surface of the wafer and uses that to calculate the extent of any surface kinetics, as well as the resulting consumption and generation of species. After repeating this step for multiple timesteps, the multiscale simulation offers a comprehensive understanding of how the wafer surface reactions evolve as time progresses inside the reactor. With the computational resources and numerical simulation specifications stated above, the computing time ranged

from 4 to 6 hours for a process time of 3.5 seconds for the 48-core and 36-core nodes, which allows for effective data production.

2.5 Multiscale Simulation Results and Discussion

The aim of the multiscale CFD simulation is to give a quantitative understanding of how various reactor designs affect the process efficiency of the AS-ALD reactions, which is directly tied to the process time. Due to the self-limiting nature of AS-ALD, the process is naturally resistant to over-deposition. Rather, the only drawbacks of overprocessing are unneeded consumption of the reagents and a decreased product throughput. Thus, there is a large economic incentive to minimize the process time as this will both reduce the reagent consumption and increase production capabilities.

To evaluate the overall process efficiency, two main criteria are taken into account: the minimum process time and surface coverage uniformity. The minimum process time is the time at which the substrate achieves full coverage for the surface-terminated product, which is directly dependent on the reagent dosage time. On the other hand, the surface coverage uniformity is a measure of how the standard deviation of the coverage on the substrate surface changes with time. Naturally, the lower the overall standard deviation, the less reagent is wasted, and the reverse implies greater reagent wastage. These two criteria are positively correlated, as a low standard deviation distribution implies an effective usage of the reagent, which then implies that the process will be completed quickly.

Fig. 2.6 illustrates the temporal progression of coverage for Steps A, B, and C for each reactor configuration. Generally, the ring-shaped reactor geometry under-performs due to the vacuum pressure forces and steric screening effects from byproducts that result in a lack of fluid transport toward the center of the wafer. Meanwhile, the single and multi-shaped injection plates are conducive towards achieving full surface coverage in minimal processing times by concentrating

the reagent toward the center of the wafer. With smaller characteristic lengths, especially for the combined reactor geometry, the injection flow rate is unable to overcome the effects of the vacuum pressure forces. Thus, an initial delay in coverage is observed for the coverage profiles of the combined model when compared to those of the single and multi feed reactors. Table 2.3 also summarizes the processing times required to achieve full surface coverage for each reactor configuration and each step in the AS-ALD ABC cycle, where the single and multi reactor models were observed to have the fastest processing times.

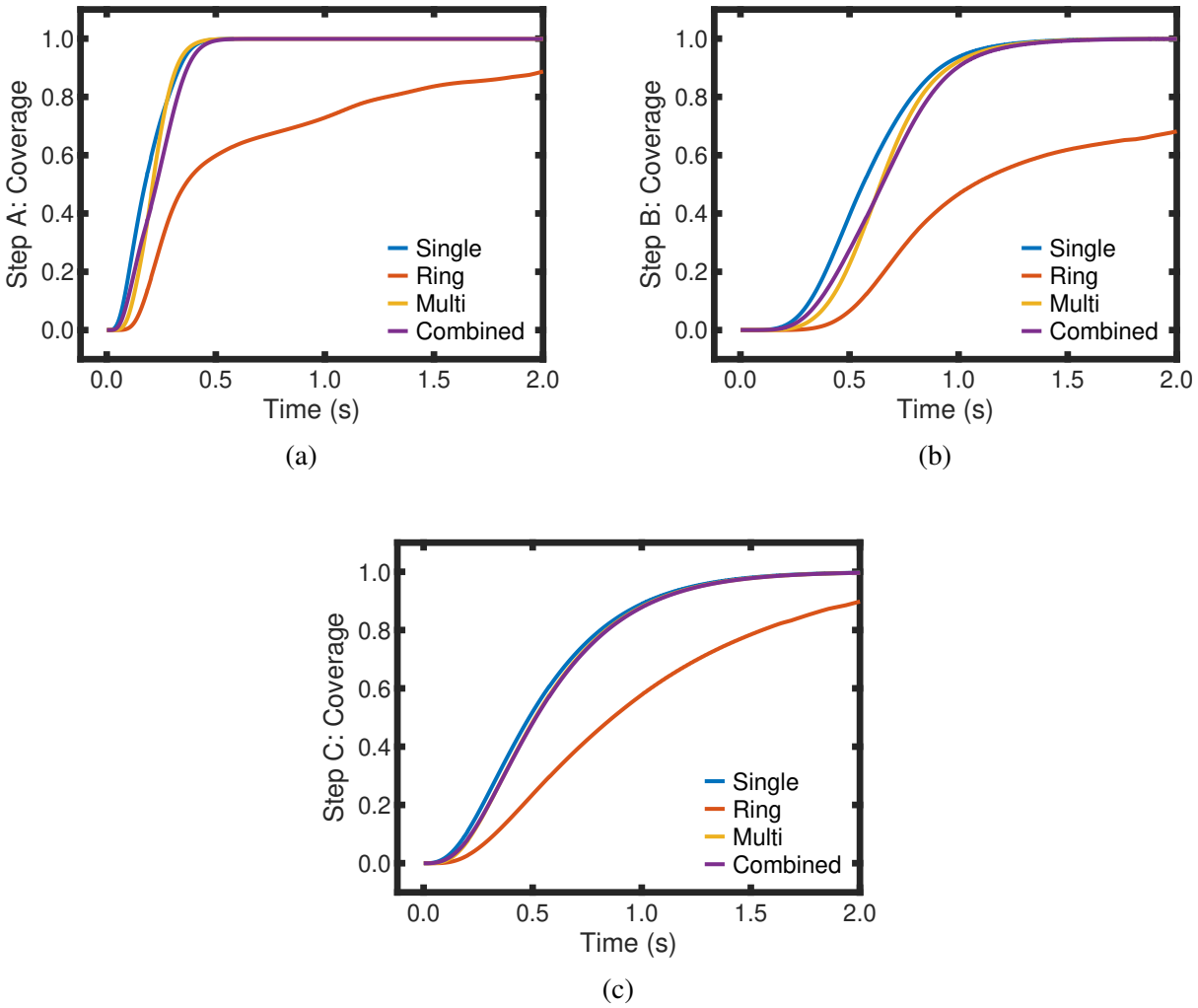


Figure 2.6: Reactor configuration comparison of the temporal progression of the average surface coverage for (a) Step A, (b) Step B, and (c) Step C.

Table 2.3: Process times required to obtain full surface coverage .

Reactor	Process Time (s)			Total Process Time (s)
	Step A	Step B	Step C	
Single	0.466	1.431	1.783	3.680
Ring	2.310	4.586	2.596	9.492
Multi	0.453	1.448	1.737	3.638
Combined	0.509	1.508	1.755	3.772

The reactor design with the smallest minimum process time can be quickly determined by examining the average coverage progression across the wafer as a function of time, as shown in Fig. 2.6. From Table 2.3, it can be seen that the multi-shaped reactor performed the best overall with a total minimum process time of 3.638 *s*. However, to understand why this reactor design performed the best, it is necessary to examine more specialized data, such as the standard deviation progression as a function of time and contour plots of the coverage at specific points in time.

The multi-shaped reactor, which has the smallest process times, also consistently has the lowest standard deviation for all points in time. However, the single and combined-shaped reactors have similar standard deviation curves for all three reactions, but the single-shaped reactor consistently outperforms the combined-shaped reactor in terms of process time. This shows that while a well-designed reactor implies a low standard deviation curve, the reverse is not necessarily true. Thus, it is not a good criterion to focus on when designing reactors.

Meanwhile, the contour coverage plots directly reflect the progress of the surface reactions. From Eqs. (2.1) and (2.2), it can be seen that, in an isothermal reactor, the reaction rates are only a function of pressure. Thus, the progress of the surface reaction at a particular point is related to the pressure which that particular point has experienced since the start of the reaction. This means that the contour coverage plots effectively represent how ideal the pressure distribution of a given reactor design is. In Fig. 2.8, all the reactor designs have a high coverage value in the

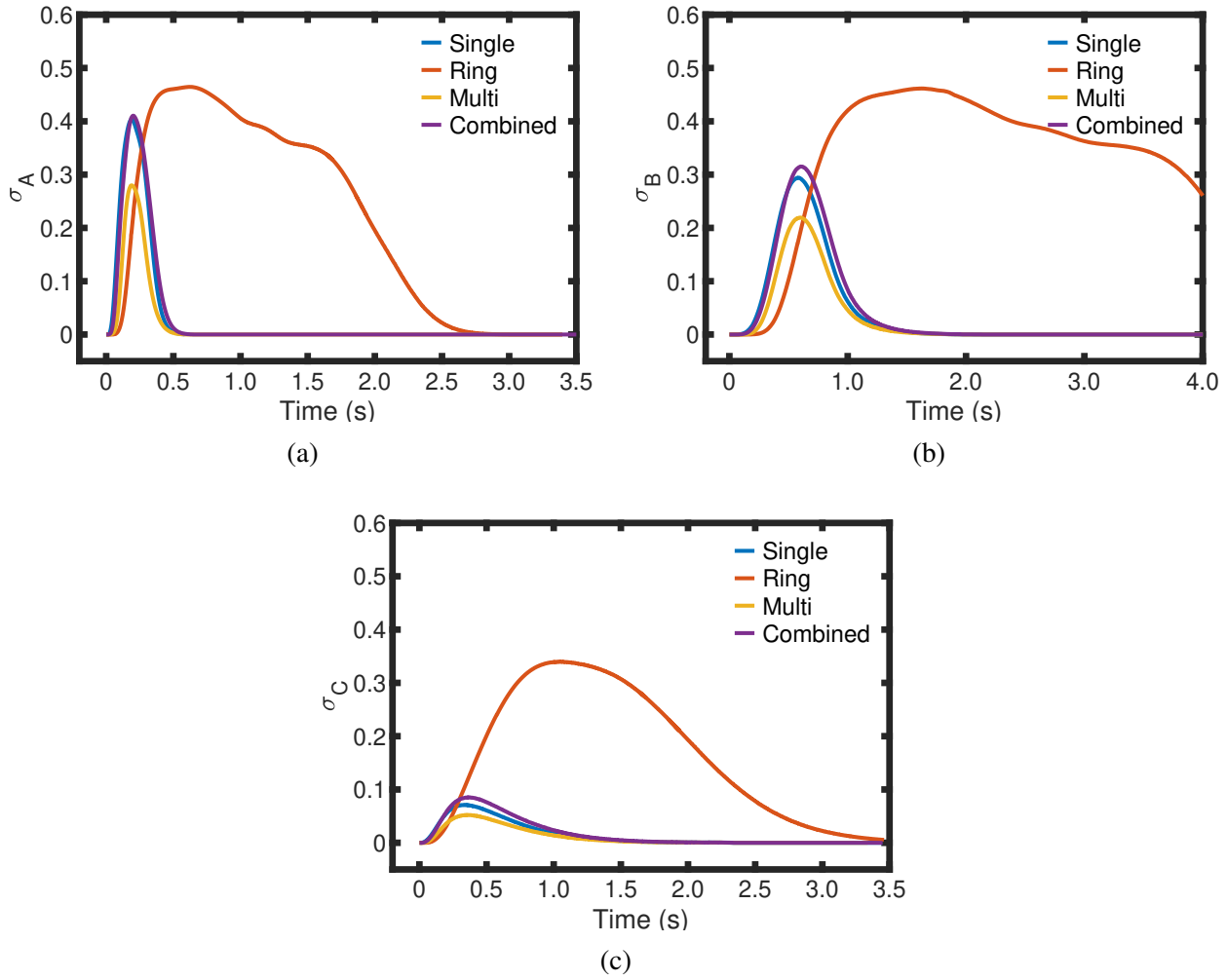


Figure 2.7: Reactor configuration comparison of the temporal progression of the standard deviation, σ, in surface coverage for (a) Step A, (b) Step B, and (c) Step C.

areas directly underneath the precursor dispensers. However, the most valuable information from these contour graphs are the areas with low coverages; these areas represent sections far from the inlet, where the reactor design plays a major role in how high the precursor pressure necessary for carrying out the reactions is. For example, the ring-shaped reactor design has a large area in the center with 0 coverage, which means that almost no precursor is flowing there. This matches the results in [83], which found that the pressure in the center is very low for the ring-shaped reactor. Meanwhile, out of the single, multi, and combined-shaped reactors, the multi-shaped reactor has

the best coverage contour plot in that its outer edges have the highest coverage. Even though the coverage at the outer edge is the same for both the single and combined-shaped reactors, the coverage at the middle of the reactor, $r = (12, 16)\ cm$, is higher for the single-shaped reactor. This means that the single-shaped reactor design has a better pressure distribution in the middle section of the wafer, rendering it superior to the combined-shaped reactor design. The results of the contour plots show that the multi-shaped reactor design is the best, followed by the single, combined, and the ring-shaped in that order. Similarly, the contour plots for steps B and C, as shown in Figs. 2.9 and 2.10, demonstrate that the multi-shaped reactor performed the best with the other three reactor designs following in the order described above. This correlates with the performances noted in Table 2.3, which shows that contour plots are a good metric for identifying efficient reactor designs. Specifically, they are able to compare and determine what reactor design has a better transient pressure profile.

It is also important to point out that the contour plots provide an important insight; there is a significant difference between the developed pressure profiles at 3 s and the initial pressure profiles for each reactor design. The former pressure profiles were examined in a previous work, which concluded that the combined-shaped reactor was the best reactor due to its optimal pressure fields [83]. However, the results presented in Table 2.3 differ substantially, as they show that the multi-shaped reactor is the best reactor with the shortest minimum process time. One explanation that accounts for both of these observances is that the pressure fields at the initial stages of the process play a pivotal role in the overall efficacy of the reactor design. The coverage evolution at the fringes of the substrate, which plays the greatest role in determining the minimum process time, is a function of the entire pressure profile evolution. Thus, it is important for the ideal reactor design to also quickly reach these fringe areas. While the combined-shaped reactor may have more even pressure profiles, the multi-shaped reactor evidently distributes the reagent quicker.

The minimum process time for each reactor design is defined as the time required to reach 99.9% coverage for each individual reaction. Overall, the multi-shaped reactor has the best results

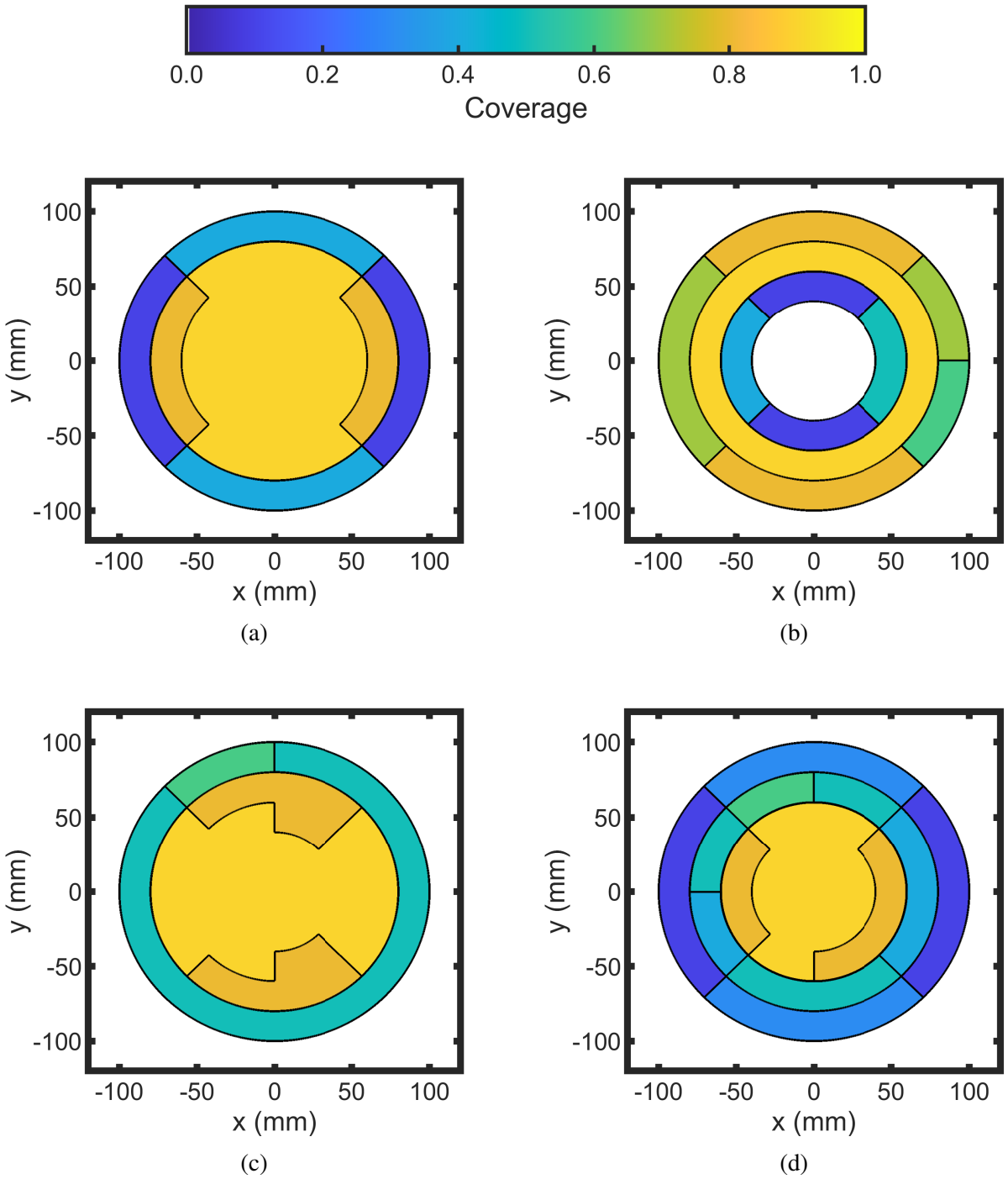


Figure 2.8: Comparison of contour plots of various reactor configurations, (a) Single, (b) Ring, (c) Multi, and (d) Combined, to study the spatial behavior of the surface coverage of the terminated Step A product for a 40-partitioned substrate at a time of 0.3 *s*.

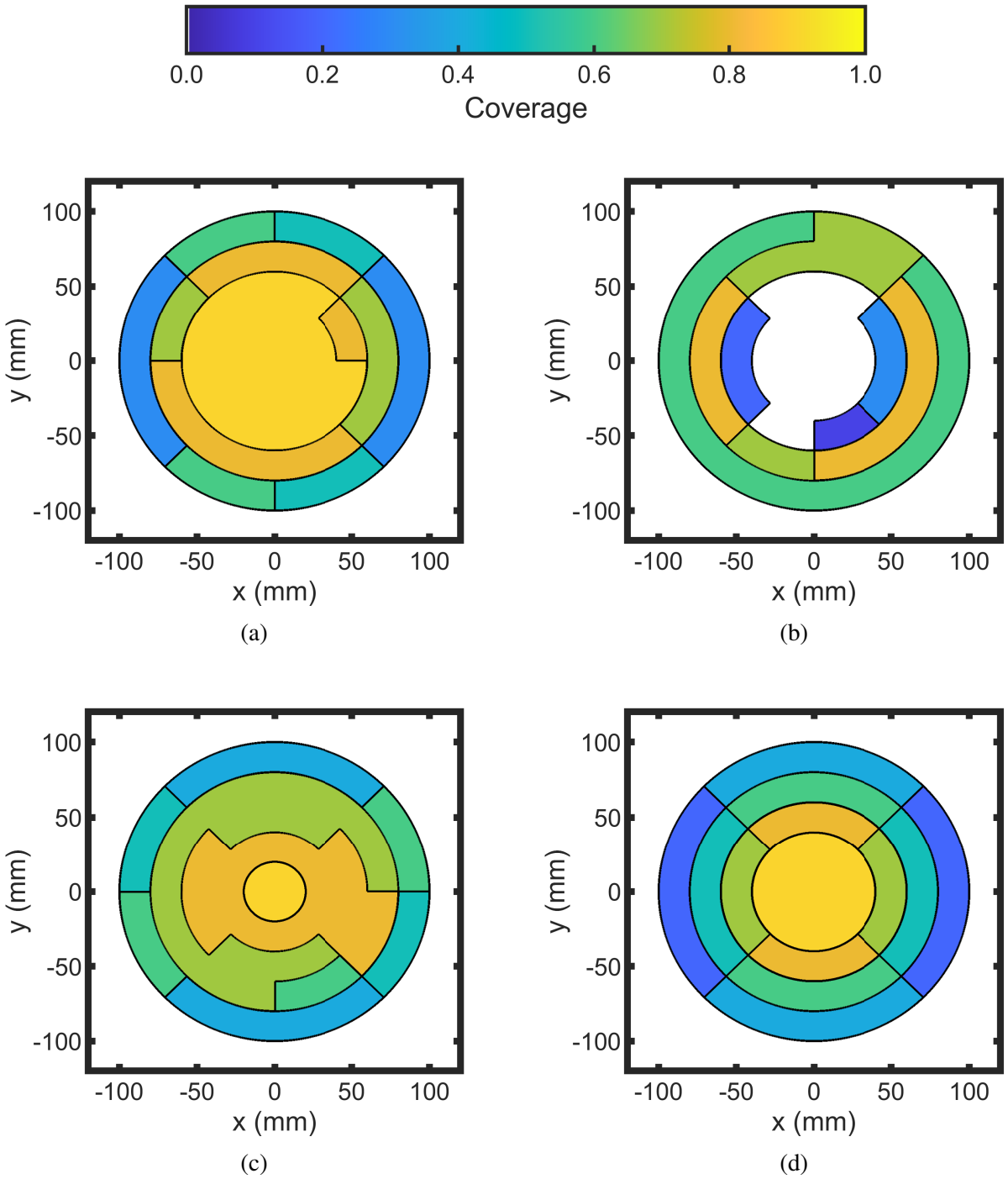


Figure 2.9: Comparison of contour plots of various reactor configurations, (a) Single, (b) Ring, (c) Multi, and (d) Combined, to study the spatial behavior of the surface coverage of the terminated Step B product for a 40-partitioned substrate at a time of 0.8 *s*.

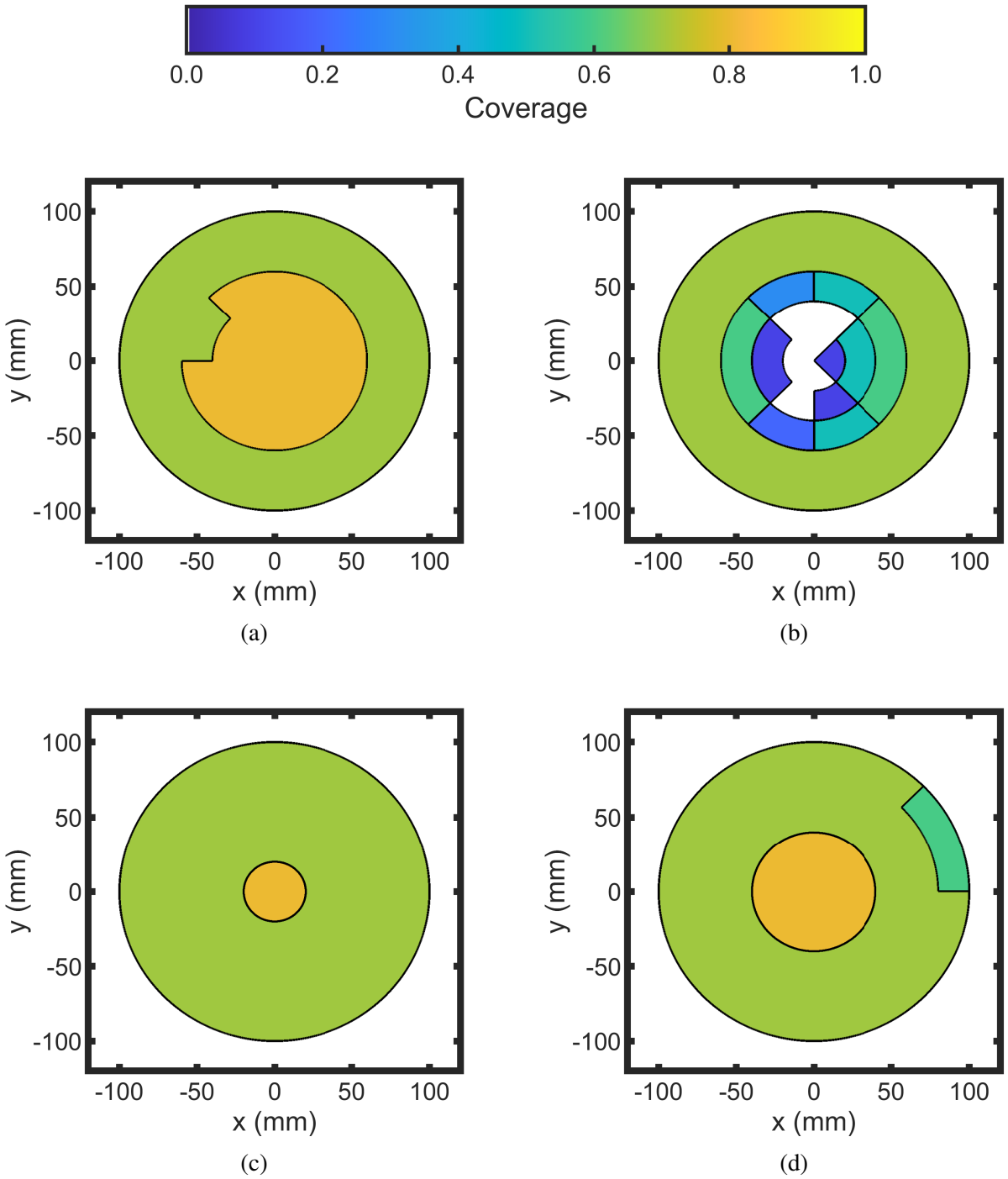


Figure 2.10: Comparison of contour plots of various reactor configurations, (a) Single, (b) Ring, (c) Multi, and (d) Combined, to study the spatial behavior of the surface coverage of the terminated Step C product for a 40-partitioned substrate at a time of 1.0 *s*.

with a total minimum process time of 3.638 s. To understand why this reactor design performs the best, the contour plot is instrumental in revealing the fact that this design spread the precursor to the remote parts of the wafer the quickest out of the examined designs. This detail explains why, even though the combined-shaped reactor has the most even pressure profiles, it ranks third in terms of minimum process time.

2.6 Conclusions

In this chapter, a multiscale computational fluid dynamics (CFD) model of an area-selective atomic layer deposition (AS-ALD) reactor was developed to study the spatiotemporal progression of surface coverage to allow the scale-up of AS-ALD processes. To improve the accuracy of the multiscale CFD model, this chapter expanded on the mesoscopic, kinetic Monte Carlo (kMC) algorithm by considering the role of the number of unoccupied atomic sites on the time progression computation. Findings suggested that surface coverage temporal progressions were similar to results conducted in prior work and were able to resolve the ambiguity of surface coverage from prior kMC methodology. Additionally, various reactor injection geometries for the macroscopic CFD simulation were constructed to enhance the mass transfer of reagent on the surface of a semiconductor substrate and study their impact on the process time required to obtain full surface coverage and the uniformity of the surface coverage with respect to time. Results indicated that two reactor models, the multi and the single, required minimal processing time and were characterized by homogeneous, temporal surface coverage.

Chapter 3

Data-Driven Machine Learning Predictor Model for Optimal Operation of a Thermal Atomic Layer Etching Reactor

3.1 Introduction

The past two decades has observed technological innovations in high-performance electronics that possess favorable characteristics including reduced feature sizes, computing speed, and energy efficiency [44]. However, this continued advancement is facing hurdles attributed to a vital and necessary component of electronic devices: semiconductors. Per Moore's Law, a semiconductor chip will improve in performance by strategic designing and downscaling of transistors that facilitate stacking and densification [2]; however, the manufacturing of these products is time consuming and inconsistent at maintaining overall quality in nanoscale dimensions. In terms of productivity, the overall semiconductor fabrication process comprises around 500 processing steps from raw materials to the finished product that are exclusive to the product material and design [79]. Additionally, some components of the finished product have stringent design criteria [60]. For ex-

ample, the nanowire, a critical component of transistors that enables current transport between the source and drain of complementary metal-oxide transistor (CMOS) technology [3, 111], demands oxide film thicknesses in the nanoscale. These nanoscale dimensions are reproducible through sequential cycles of thin-layer deposition and etching processes that are intended to add or remove monolayers of substrate material by exhibiting self-limiting characteristics. However, it is difficult to streamline a high quality fabrication method for this process at an industrial setting due to the inherent limitations in process- and operation-dependent techniques. One way to help meet the growing demand for both semiconductor chip throughput and quality is by developing models that can accurately predict the process quality as a function of real-time process data.

Digital twin modeling is an effective tool intended to replicate real-world processes through computational modeling with feedback validation to ensure the efficacy of the model. For example, Shao et al. (2019) discussed the prevalence of digital twin modeling in the semiconductor manufacturing industry with applications to smart manufacturing, which enables process optimization and advanced process control by extending data sets across wider ranges of operating conditions [74]. Besides a single process, Moyne et al. (2020) hypothesized the potential for expanding beyond a singular process (i.e., across processes with slightly dissimilar characteristics) through simulated modeling [55]. Kanarik et al. (2023) also entertained the idea of implementing artificial intelligence to construct data-driven and predictive models to enable process optimization [35]. The aforementioned works discuss clever approaches for integrating a network of simulated models that is applicable for this study.

With the growing complexity of semiconductor manufacturing processes, manually measuring wafer quality has become increasingly time-consuming and labor-intensive [51]. For instance, a quartz crystal microbalance is traditionally employed to measure the thickness of deposited or etched films on wafers in industry, thereby assessing the etch coverage and quality of the product in an off-line analysis manner. However, the use of a quartz microbalance requires careful operation, bears a low sampling rate, and has significant expenses [33]. To address these challenges, soft

sensing methods have emerged in recent years [32]. A soft sensor functions like a traditional sensor by utilizing models that interpret process data that is easier and less expensive to obtain, to predict product properties such as the etch coverage across the wafer. The performance of a soft sensor is highly dependent on the accuracy of its prediction model. Deep learning methods, including Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), and Transformers, have gathered significant attention for this purpose due to their superior performance and wide applications [81].

In industry, large fabrication plants often have multiple product flows that share the same etch process [6]. For example, multiple product flows that each create a specific semiconductor device may all use the same 1000 Å Al_2O_3/SiO_2 etch process. Subtle differences between these product flows, such as the underlying substrate geometry, often cause their respective substrates to exhibit different kinetic behaviors for the same process recipe. Thus, a predictor model trained on one specific product line and reactor is not guaranteed to perform similarly for other product lines or reactors, which is why most such predictor models thus far have been focused on a specific process on a specific tool [106].

However, with the increasing demand for more semiconductor products, fabrication plants must output numerous semiconductor chips. This has led to an increased number of both processing tools and process flows [12]. As the number of tools and flows in a fabrication plant increases, the number of predictor models needed will increase dramatically, leading to numerous unique process datasets. Thus, it is important to explore if data aggregation techniques can be used to create a general predictor model that performs well for a set of process flows. And as the number of unique process datasets continues to increase, it will also become important to create a methodology to determine which datasets should be aggregated for the training of a predictor model.

Modern ALE processes operate quickly, taking less than 10 minutes to complete a single cycle [34]. However, to capture the inherent process variability for a specific product flow and reactor, process data must be collected from multiple runs. Thus, if the required process data were to be

gathered experimentally at an academic scale, it may take months or even years to collect enough data to train an accurate predictor model. An alternative method is to use simulated process data. Simulated process data offers the convenience of being able to directly manipulate kinetic parameters, which drastically decreases the number of runs required to collect enough process data to train a predictor model. With simulated data, valuable insights into the efficacy and optimization of data aggregation techniques within the purview of creating predictor models for semiconductor manufacturing processes can be found.

This work investigates the development of a cross-operation and cross-design predictive model of a two-step thermal ALE procedure to fabricate an aluminum oxide (Al_2O_3) film by utilizing a precursor trimethylaluminum (TMA) and an etching reagent hydrogen fluoride (HF) under high operating temperatures that volatize all the reactants and products. This process is simulated by a computational model with a multiscale framework that intersects features from various time and length domains for atomic, molecular, kinetic, and fluidic properties, all of which govern the kinetics of this etching process. In the Ångstrom and picosecond length and time scales, respectively, atomistic modeling is relevant for discussing the electronic, thermophysical, and kinetic properties of the materials involved in the ALE process by employing ab initio molecular dynamics simulations such as density functional theory (DFT) and nudged elastic band (NEB) methods. Additionally, the spontaneity or tendency for a reaction to occur is characterized through elementary reaction pathways that are defined using a statistical mechanics via collision theory (CT) and the Arrhenius model. To establish this stochastic behavior of reaction in a larger length and time scales of micrometers and milliseconds, respectively, a mesoscopic approach through a kinetic Monte Carlo (kMC) method is beneficial for this purpose. Lastly, ALE is dependent on the fluid dynamics when the substrate is exposed to reagents and byproducts that affect the composition and characterization of the substrate surface. Computational fluid dynamics (CFD) is utilized to study the effects of fluid transfer on the substrate surface, in which CFD is not bounded by a maximum length- and timescale exemplified by atomistic and mesoscopic models. The conjunction

of these three simulations establish a multiscale model that can (1) resemble realistic experiments conducted in vitro, (2) produce synthetic data at an efficient rate, and (3) enable optimization of process operation and design by conducting numerous case studies for similar systems defined by quantifiable variables.

Traditional model reduction methods like proper orthogonal decomposition and approximate inertial manifold techniques work well for the construction of low-dimensional models for transport-reaction processes modeled by one- or two-dimensional parabolic partial differential equations (PDEs) and can lead to low-order models for controller design; however, this chapter deals with the development of input/output models capturing nonlinear relationships between process operating variables as inputs (e.g., input flow rates and pressure) and product metrics as outputs (e.g., film spatial uniformity, coverage) that cannot be captured by the traditional model reduction methods for PDEs [96]. Furthermore, the model of the ALE process considered in this chapter is a multiscale CFD model whose complexity makes the implementation of traditional model reduction methods very challenging given the use of different continuum (e.g., dynamic conservation equations) in the gas phase and discrete (e.g., kinetic Monte-Carlo models) on the film surface. Thus, this chapter explores using neural network models and the associated statistical analysis as an efficient way to develop predictive operational models for these complex processes.

In summary, this chapter investigates the application of a machine learning-based soft sensor for predicting ALE etch rates using process data that can be realistically collected in real-time during the process. The necessary process data were generated through multiscale simulations of an ALE process in a discrete feed reactor, wherein kinetic parameters were adjusted to represent different process flows. To enhance model performance, a data aggregation approach is proposed, which increases the volume of training data by integrating multiple datasets. This chapter is organized by first describing the overall structure of what datasets are generated and aggregated, and how they are compared. The subsequent section goes into detail regarding the multiscale modeling simulations that were used to generate process data. The following section describes how the pre-

dictor model was developed and trained. The last section analyzes the performance of the models trained on different datasets and their implications.

3.2 Process Description and Data Generation Methods

The effectiveness of data aggregation depends greatly on the data being aggregated. To that end, this chapter seeks to elucidate the possible benefits of aggregating data and explore methods of choosing which datasets should be aggregated by focusing on a specific fabrication process. The ideal process would be one that is relatively simple, has at least two sources of variance, and would benefit from a robust predictor model. One semiconductor manufacturing process that meets all three of the aforementioned criteria is Atomic Layer Etching (ALE).

Specifically, this chapter explores data aggregation in the context of using ALE to etch away Al_2O_3 while preserving the underlying SiO_2 substrate. This process is simple in that it is only composed of two half reactions, whereas other processes such as area-selective atomic layer deposition may consist of three or more [92]. The two half-reactions are shown and described below.

$$\mathrm{Al_2O_3\ (s) + 6HF\ (g) \rightarrow 2AlF_3\ (s) + 3H_2O\ (g)} \qquad \text{(Reaction A)}$$

In Reaction A, gaseous HF prepares the Al_2O_3 surface for etching.

$$\mathrm{2AlF_3\ (s) + 4Al(CH_3)_3\ (g) \rightarrow 6AlF(CH_3)_2\ (g)} \qquad \text{(Reaction B)}$$

In Reaction B, gaseous trimethylaluminum (TMA) etches away the prepared ALF_3 surface created in Reaction A, releasing a byproduct of dimethylaluminum fluoride (DMAF). These etching reactions were chosen due to their high selectivity towards Al_2O_3 rather than SiO_2. To ensure that all the reagents are in a gaseous form, the process temperature is 573 K and the process pressure is 300 Pa. Additionally, the process consists of Reaction A and Reaction B cycling back and forth;

to reduce reagents from the other half-reaction mixing, a purge step is taken before and after each reaction. Specifically, N_2 is pumped into the chamber until all reagent is removed. For Reaction A, 1e-5 kg/s of gas with a mole fraction of 0.1 HF and 0.9 N_2 is pumped into the chamber for 1.2 s and then purged with N_2 for 0.8 s. For Reaction B, 1e-5 kg/s of gas with a mole fraction of 0.5 TMA and 0.5 N_2 is pumped into the chamber for 1.5 s and then purged with N_2 for 0.5 s. From experimental testing, these conditions with no disturbances result in complete reactions for both half-reactions [100]

To simulate these reactions, both half-reactions can be decomposed into multiple intermediate reactions, as detailed in Yun et al. (2021) [100]. These intermediate reactions can be classified into two types: adsorption and nonadsorption. By varying the kinetic parameters of these intermediate reactions, variance can be naturally introduced into the process. These varied parameters represent differences in the substrates, such as the device geometry or substrate material [54]. They exist because multiple product flows and devices share process recipes along the manufacturing pipeline, and one of the goals of this chapter is to demonstrate that aggregating process data will result in improved model performance despite the fact that the data originate from different sets, each with their own unique kinetic parameters.

For the final criterion, a robust predictor model would, at a minimum, be able to improve manufacturing efficiency by eliminating inspection steps. These inspection steps are conducted at state-of-the-art fabrication centers, where each wafer is inspected after an etch step to ensure that the product specifications and quality standards are met. However, with a robust predictor model, this step could be omitted (or at least executed less frequently) for a majority of processed wafers, greatly increasing overall manufacturing efficiency.

After selecting a process, the next step is to gather data for that process. Generally, most machine learning processes improve in performance when trained on larger datasets. Thus, each data subset needs to be large enough to produce a functioning model to properly compare the effects of data aggregation. To generate all the requisite data, it would take months, if not years,

to do so with industrial data. An alternative method is to use simulations to generate the requisite data. By constructing a reactor model and using advanced simulation methods, it is possible to accurately represent an ALE reactor and the reaction kinetics occurring within it [92]. Additionally, simulation runs are much faster to execute than their physical counterparts; with powerful computer processors, it is possible to complete over 16 unique process runs in a single day, for example [84]. Thus, all the datasets analyzed in this chapter are generated through simulations.

3.2.1 Multiscale Computational Fluid Dynamics Modeling

All simulations must balance accuracy and computational efficiency while generating data at a sufficient rate. Generally speaking, the more accurate a simulation is, the more computational power it takes to carry out that simulation. Because the simulated data is meant to be used as training data for a machine learning algorithm, the simulations must be fast while sacrificing the least amount of accuracy.

To meet these requirements, this chapter carried out two-dimensional (2D) multiscale CFD simulations of an ALE system in a discrete feed reactor. The discrete feed reactor is a stationary reactor with a single inlet at the top, two outlets on the sides, the wafer substrate at the bottom, and a showerhead plate between the wafer and the inlet to promote ideal reagent distribution as shown in Fig. 3.1. The reactor operates by first purging the chamber with N_2 gas, then injecting a reagent gas into the chamber for a preset process time, and finally purging the chamber with N_2 gas once again. This configuration makes it easy to optimize reagent usage while maintaining etch uniformity, which ensures that the end product reaches process specifications [83]. The simulation was bound to 2D geometry to reduce the computational complexity and is valid because both the substrate and reactor have radial symmetry. Multiscale CFD simulations decrease computational complexity (with respect to carrying out a microscopic simulation for the entire process domain) while preserving accuracy by simultaneously carrying out macroscopic and microscopic simulations that constantly communicate between each other (Wang 2024). Specifically, the simulation,

shown in Fig. 3.1, takes place within the multiphysics software, ANSYS Fluent, which simulates the macroscopic mass and energy dynamic balances within the reactor, which is represented by the left, red section in Fig. 3.1. At each integration timestep of the gas-phase continuum model, custom user-defined functions simulate the mesoscopic reaction kinetics on the wafer surface through a kinetic Monte Carlo (kMC) simulation scheme over several locations on the wafer surface, which is represented in the right, blue section in Fig. 3.1. Additionally, the macroscopic simulation receives information regarding how much reagent is consumed and product is produced at each point on the wafer, which is the green box at the bottom of Fig. 3.1, while the kMC simulation receives information regarding the partial pressures at each point of the wafer, which is the orange box at the top of Fig. 3.1. These two simulations are carried out simultaneously through user-defined functions (UDFs) in ANSYS Fluent that enable customizable features applicable to multiscale modeling, and by working in tandem, they improve the overall accuracy of the ALE simulation.

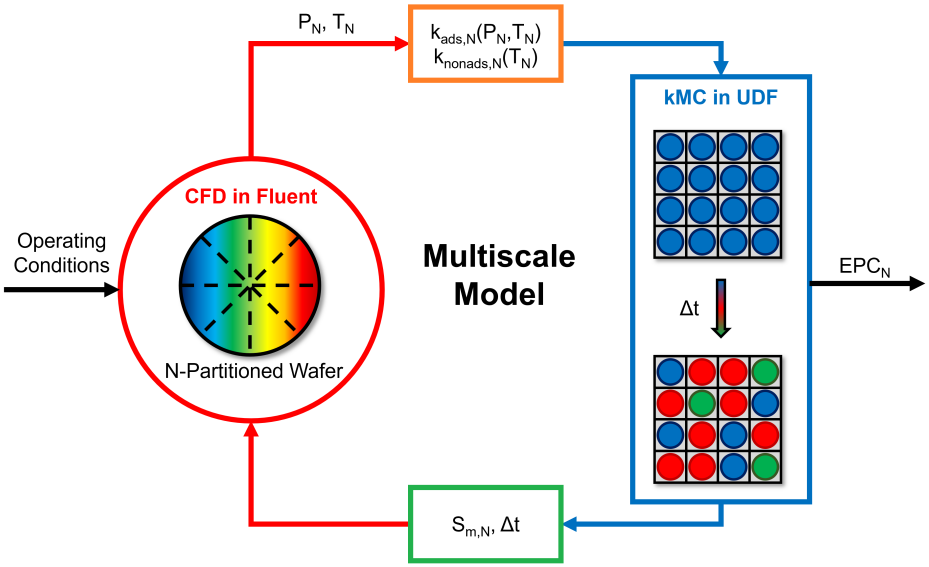


Figure 3.1: figure showing the flow of information between the macroscopic simulation in ANSYS Fluent and the mesoscopic simulation executed through the UDFs.

While the macroscopic simulation takes place inside ANSYS Fluent, the reactor was designed in ANSYS SpaceClaim, a 3D CAD software. The discrete feed reactor has a cylindrical shape

and consists of a wafer plate at the bottom, an inlet plate in the middle to ensure even reagent distribution, an inlet hole at the top for reagents to enter, and 2 outlet holes at the sides for the etch byproducts and purge gas to exit from. In a previous work, it was found that the efficacy of these reactors mainly depends on 2 factors: the gap distance and inlet plate geometry [83]. Based on previous research into reactor optimization, the reactor used in this chapter has a gap distance of 5 *mm* and an inlet plate of 13 equally spaced holes with a diameter of 10 *mm* [92]. A visual representation can be seen below.

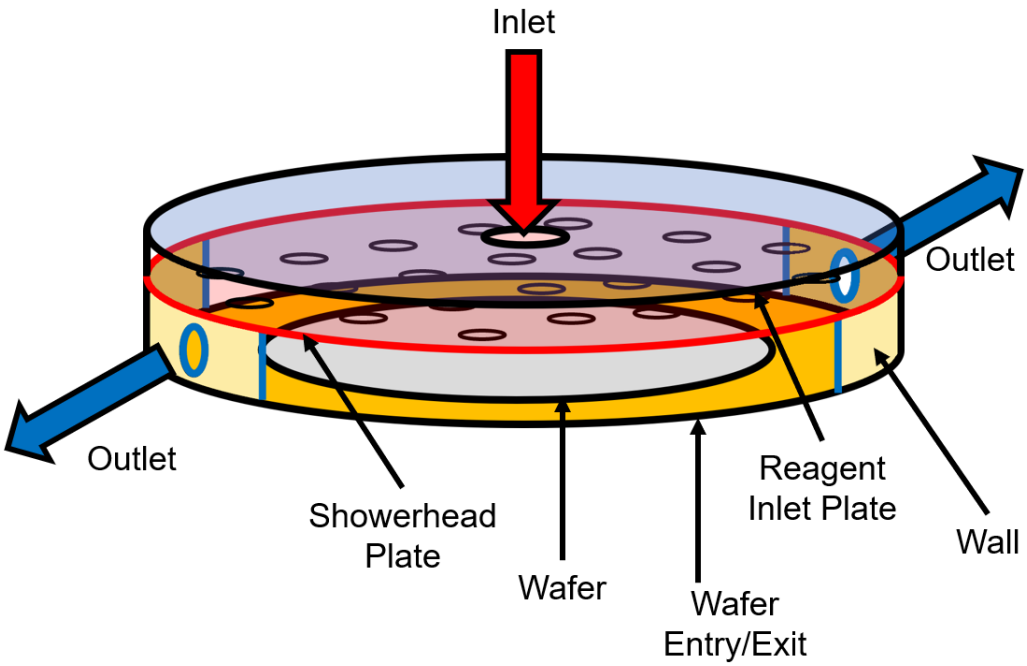


Figure 3.2: Schematic of the reactor model. For Reaction A, a mixture of HF and N_2 enters from the inlet, and HF, H_2O, and N_2 are purged through the outlet. For Reaction B, a mixture of TMA and N_2 enters from the inlet, and DMAF and N_2 are purged through the outlet.

To observe the different reaction rates across the wafer surface while minimizing excessive calculations, the wafer surface was separated into 5 equidistant sections. Each section has its own kMC simulation, which takes in the average partial pressures of that wafer section and returns the mass fluxes of that wafer section. This allows each section to progress on their own and ultimately

allows for analysis of the etching uniformity across the wafer.

The macroscopic simulation evaluates the spatiotemporal behavior of the fluid transport within the reactor by numerically solving the characteristic mass, momentum, and energy transport equations, which are respectively described as follows:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \overrightarrow{v}) = S_m \tag{3.1}$$

$$\frac{\partial}{\partial t} (\rho \overrightarrow{v}) + \rho (\overrightarrow{v} \cdot \nabla) \overrightarrow{v} = -\nabla P + \nabla \cdot (\overline{\overline{\tau}}) + \rho \overrightarrow{g} + \overrightarrow{F} \tag{3.2}$$

$$\frac{\partial}{\partial t} (\rho E) + \nabla (\overrightarrow{v} (\rho E + P)) = -\nabla \left(\Sigma h_j \overrightarrow{J}_j \right) + S_h \tag{3.3}$$

where ρ is the gas-phase species density, $\overrightarrow{v}$ is the velocity of said species, S_m is the source generation and consumption flux of that species, P is the operating pressure of the reactor, $\overline{\overline{\tau}}$ is the normal two-rank stress tensor, $\overrightarrow{g}$ is the gravitational acceleration constant, $\overrightarrow{F}$ is the force acting on the system, E is the accumulated rate of system energy, S_h is the energy source generation or consumption, h_j is the sensible enthalpy flux of gas species j, and $\overrightarrow{J}_j$ is the mass diffusion flux of gas species j. Eqs. (3.1) to (3.3) are numerically solved at each integration timestep of 0.001 s. The integration time step used for the solution of the continuum gas-phase dynamic model, 0.001 s, is small enough to ensure stable numerical integration and high simulation accuracy while keeping the computational burden at an affordable cost. Smaller integration time steps have been tested, and they lead to the same numerical results even though the computational burden is increased. Increasing the integration time step to higher values will speed up the calculations but may compromise numerical stability and reduce accuracy, and as the computational burden under the 0.001 s time step is computationally affordable, there is no reason to do so. Clearly, there is an upper bound on the time step that maintains numerical stability of the simulation, but 0.001 s is well within this upper bound.

The only reactions occurring within the reactor take place on the wafer surface. Thus, all

boundaries are defined as impermeable walls, save for a few exceptions: the inlet hole at the top is defined as an inlet with a mass flowrate of 1e-5 kg/s, the outlet holes at the sides are defined as outlets with a dynamic pressure of -200 Pa, and the wafer surface is defined as a reaction zone where the molecular species flux calculated in the kMC section are implemented. More details about the macroscopic simulation can be found in [83].

The mesoscopic simulation recives the partial pressures calculated in the macroscopic simulation and returns the mass flux of products and reactants. This computation is conducted through a kMC algorithm that simulates the reaction rates of the nonadsorption reactions on the wafer as a function of the kinetic rate constants. For this ALE process, two general types of reactions are considered: adsorption/desorption reactions and nonadsorption reactions. The rate constant of the former is calculated through Collision Theory, and that of the latter is calculated with the Arrhenius equation, which are respectively shown below.

$$k_{ads,d} = \frac{2P_d A_{site}\sigma}{Z_d\sqrt{2\pi m_d k_B T}} \tag{3.4}$$

where $k_{ads,d}$ is the reaction rate constant for gas species d to adsorb onto the wafer surface, P_d is the partial pressure of gas species d, A_{site} is the surface area of the binding site, σ_s is the experimentally determined sticking coefficient of gas species d, Z is the coordination number of gas species d, m_d is the atomic mass of gas species d, k_B is the Boltzmann constant, and T is the absolute temperature of the wafer surface.

$$k_{nads} = \nu \exp\left(\frac{-E_a}{RT}\right),\ \nu = \frac{k_B T Q^{\ddagger}}{hQ} \tag{3.5}$$

where k_{nads} is the reaction rate constant for a nonadsorption reaction, ν is the pre-exponential factor, E_a is the activation energy, R is the universal gas constant, T is the absolute temperature of the wafer surface, and h is the Planck constant. Note that when calculating ν, the ratio of the partition functions $Q^{\ddagger}, Q$ is assumed to be unity [31]. This assumption was validated by comparing the

resulting simulation results to experimental results. Additionally, all the other kinetic properties, such as σ, E_a, and ν, for both half-reactions of the ALE process were also determined in that same work [100].

The kMC algorithm uses a grid with 300×300 sites to represent a larger swath of the wafer surface and randomly generated numbers to represent the stochastic nature of the ALE half-reactions. In a previous work, it was found that this grid size allowed for accurate simulations for a low computational cost [100]. At each integration timestep of the macroscopic CFD simulation, the kMC algorithm calculates the rate constants of each possible reaction with Eqs. (3.4) and (3.5) and then uses them to update the 300×300 grid.

The basic steps of the algorithm are as follows:

1. Randomly select a site on the grid.
2. Randomly select 2 random numbers, $\gamma_1, \gamma_2 \in (0, 1]$.
3. Sum up all possible reaction rate constants into k_{tot}.
4. Select a reaction by comparing γ_1 to k_{tot}.
5. Calculate the time evolution as $\delta t = -\ln(\gamma_2)/(nk_{tot})$, where n is the number of active sites on the grid.
6. Terminate when $(\sum \delta t) \geq t_{int}$, where t_{int} is the integration timestep of 0.001 s.

Note that both γ_1 and γ_2 are randomly selected from a uniform distribution of $(0, 1]$ as they represent the stochastic nature of atomistic surface reactions. Additionally, For Step 3, the reaction rate constants are calculated with Eqs. (3.4) and (3.5) and the atomistic constants found in [100]. Eq. (3.4) is dependent on the partial pressure of the reactant species, which means that all the reaction rate constants must be recalculated at each time step as the partial pressures of each species cannot be guaranteed to be constant.

Steps 1, 2, 3, 5, and 6 are relatively simple, but Step 4 requires a more in-depth explanation. To select a reaction, the following expression is evaluated for each possible reaction:

$$\sum_{i=1}^{r-1} k_i < \gamma_1 k_{tot} \leq \sum_{i=1}^{r} k_i \tag{3.6}$$

where r is the reaction the expression is being evaluated for, k_i is the ith reaction, γ_1 is a randomly selected number created in Step 2, and k_{tot} is as described in Step 3. Additionally, for all grid sites and a selected γ_1, Eq. (3.6) will only be true for a single, unique reaction, which is the selected reaction. A more detailed explanation of the kMC algorithm, including pseudocode, can be found in a previous work [92].

At the end of the multiscale simulation, the reaction coverage across the wafer can be estimated by examining the 300×300 of the kMC simulation. As the coverage is a fraction of how much of the surface has completed the reaction, it is simply the number of sites that have reached the final product divided by the total number of grid sites. The coverage can then be converted into the etch rate by multiplying by 0.46 Å/cycle, as that is the amount of Al_2O_3 that would be etched away if the wafer reached full coverage in both half-reactions [100].

With the simulation settings described above, the accuracy of the simulations is uncompromised while the computational efficiency remains high. When run on powerful CPU-based (central processing unit) nodes with 24 and 48 cores with 384 GB and 512 GB of DRAM (dynamic random-access memory), respectively, a full simulation of both half-reactions took approximately 8 hours. Thus, by using 4 such nodes, simulating 100 process runs can be completed in 200 hours.

3.2.2 Process Datasets

For this project, two types of datasets were created: one that represents a process, and one that imitates raw industrial data. The main difference between these two dataset types is that the former datasets are meant to be compact datasets that represent many years of process data. Their

Table 3.1: Kinetic parameter ranges for each process.

	f_σ	f_ν
CT	[0.5,1.5]	—
TST	—	[0.5,1.5]
MIX	[0.5,1.3]	[0.5,1.5]
INV	[0.5,1.3]	[1.5,0.5]

kinetic parameters are set to a specific number within a range and do not include any noise; thus, they will be called process-specified datasets. The latter dataset, which is used to compare model performances, is intended to represent a series of process runs. The kinetic parameters for each run are randomly selected from a Gaussian distribution that represents a specific process, which introduces variation into each process run. Thus, this dataset will be called the random-run dataset.

To train and validate the predictor models, four separate process-specified datasets with varying kinetic parameters were created. The varied kinetic parameters were chosen to represent common process shifts in a manufacturing environment, such as deposition of reactants to reactor sidewalls affecting the nonadsorption reactions or complex device geometries affecting sticking coefficients [97].

The four process datasets are differentiated by their kinetic constants, which have been uniquely modified for each dataset. Specifically, the sticking coefficients (σ) and pre-exponential factors (ν), are multiplied by a constant ranging from 0.5 to 1.5. The four processes are listed in Table 3.1. where f_σ is the constant that the sticking coefficient (σ) found in Eq. (3.4) is multiplied by and f_ν is the constant for the pre-exponential factor (ν) found in Eq. (3.5). Furthermore, in CT, only f_σ is varied; in TST, only f_ν is varied; in MIX, f_σ and f_ν share the same value; finally, in INV, f_σ and f_ν are inversely correlated such that their sum is always 2. For example, the dataset that represents MIX consists of 25 simulations, where the values of f_ν and f_σ are represented as:

$$f_{\nu,i} = f_{\sigma,i} = 0.47 + 0.03 \cdot i$$

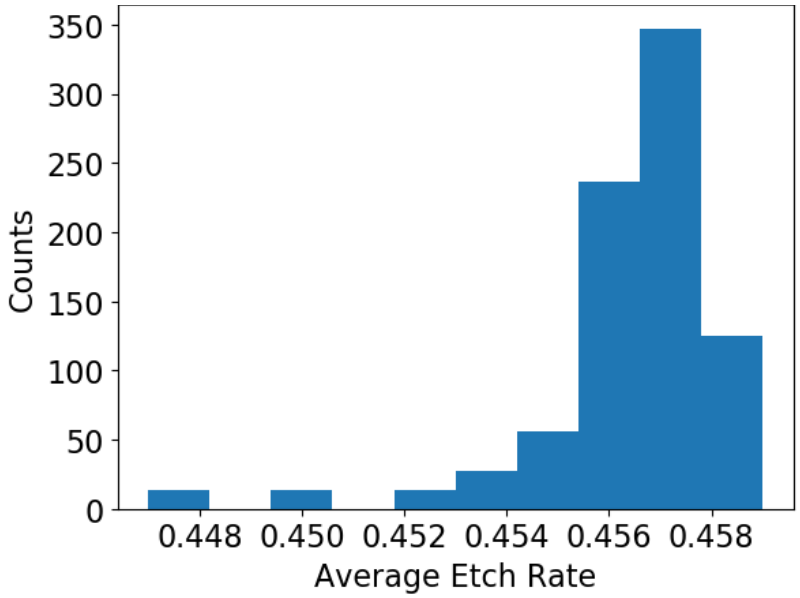


Figure 3.3: Histogram of the average etch rate across the entire wafer for each run in the random-run dataset.

where i is the i^{th} simulation. By carrying out a similar procedure for the CT, TST, and INV processes, 4 unique datasets comprising 25 simulations each are generated, making for a total of 100 simulations. These datasets are then aggregated in various combinations before being used to generate a prediction model.

To analyze the performance of the prediction models generated with the process-specified datasets, each model will be used to estimate the etch rate of a random-run dataset comprising 60 process runs. The kinetic parameters for each run will be randomly selected from a Gaussian distribution that has an average of 1 and a standard deviation of 0.1. To consider both the average etch rate and the standard deviation of the etch rate across the wafer, the etch rate is measured at 5 inspection points across the wafer that correspond to the 5 wafer sections described in the "Multi-scale Modeling" section. The resulting etch rate mean and spread for the random-run dataset are illustrated in Figs. 3.3 and 3.4, respectively.

Note that the etch rate distribution is not Gaussian, as the process metric of etch rate does not vary linearly with respect to the kinetic parameters. As both ALE half-reactions are naturally self-

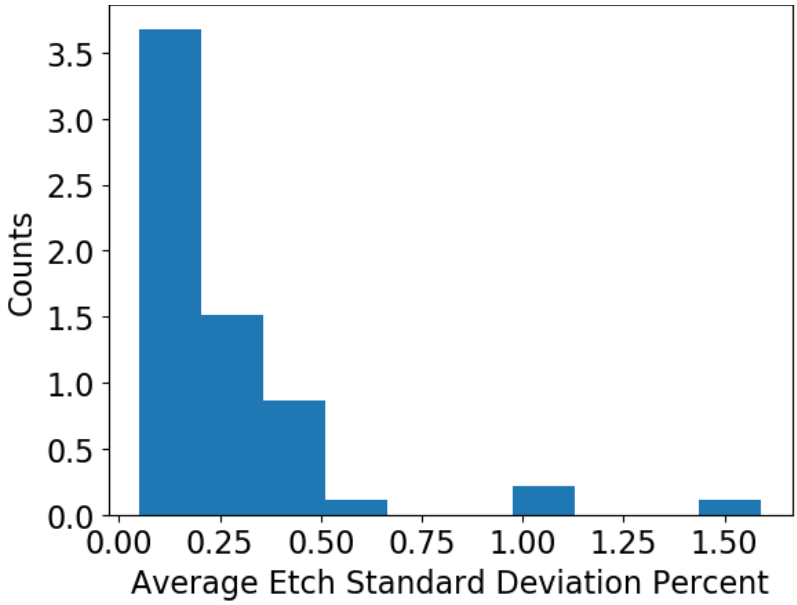


Figure 3.4: Histogram of the standard deviation of the etch rate across the wafer as a percentage of the average etch rate for each run in the random-run dataset.

limiting, high kinetic parameters that increase the reaction rate of each half-reaction will minimally impact the overall etch rate [100]. Thus, the etch rate distribution is skewed.

Besides measuring the performance of each model as a predictor, it is also possible to convert them to classifiers and measure their efficacy in that regard. This approach is accomplished by adding a filter to both the random-run dataset and the model output that analyzes the etch rate mean and standard deviation and classifies it as either a "Pass" or "Fail." The specific pass/fail metrics were chosen such that the fail rate is around 2% for a process with kinetic parameters with a mean of 1 and standard deviation of 0.1. This means that, for the etch rate mean and etch rate standard deviation, if either process variable is more than 2 standard deviations from the mean, the run has failed. For the random-run dataset, this means that a run fails if the etch rate mean is less than 0.444 or if the etch rate standard deviation is greater than 2% of the mean. When applied to the random-run dataset, these criteria result in a fail rate of 2.3%. This fail rate is consistent with industrial datasets the authors have worked with.

3.3 Transformer Model Training Method

The optimal model training method will differ for each process, depending on what process data are readily available and able to be collated into datasets. For the purposes of this chapter, we aim to train a transformer model that takes real-time process data as an input and outputs the expected etch rate.

3.3.1 Input Variables for Model Training

For an ALE reactor operating in an industrial environment, pressure and temperature are critical process variables that can be measured at high frequencies with low latency, enabling the generation of comprehensive time-series datasets for process modeling. To ensure consistency and simplicity, the wafer temperature and reactor operating pressure are maintained at near constant values by feedback control systems. Then, the surface pressure of the wafer, measured in Pascals, is recorded every 0.1 s, which aligns with the limitations of real-life industrial sensors. However, to prevent information loss, because the simulation is conducted with an integration timestep of 0.001 s, the pressure data is also extracted at every integration timestep. Prior studies on the ALE of Al_2O_3 in discrete feed reactors indicate that a 2 s period is sufficient for achieving full wafer coverage under most conditions [83]. Consequently, the pressure measurement period is limited to 2 s for each half-cycle, resulting in a maximum of 20 points of time-series wafer surface pressure data per half-cycle. Each data point contains 5 pressure readings, which are sampled from the 5 different inspection points on the wafer surface described in the "Multiscale Modeling" section. The time-series pressure data collected for both the HF and TMA half-cycles are then concatenated into a complete dataset for a single ALE run.

3.3.2 Output Variables for Model Training

The output variable of the ALE process for the soft sensor is the etch coverage calculated for each of the previously described inspection points at the end of the overall process. This measurement offers valuable information about both the overall etch rate and the etch rate uniformity across the wafer. The total coverage after two half-cycles is defined by:

$$cov_i = cov_{\mathrm{HF},i} \cdot cov_{\mathrm{TMA},i},\ i \in \{1, 2, 3, 4, 5\} \tag{3.7}$$

where cov_i is the total coverage at the end of the process on inspection point i, $cov_{\mathrm{HF},i}$ is the coverage at the end of HF half-cycle on inspection point i, and $cov_{\mathrm{TMA},i}$ is the coverage at the end of TMA half-cycle on inspection point i. Each half-cycle is programmed to end at $t = 2.0\ s$, where t is the processing time.

The true total coverages are obtained through multiscale simulations as described in the previous section and then compiled into datasets consisting of their respective input and output data. When effectively trained, the soft sensor model is expected to output the coverage data with minimal absolute percentage error compared to the true values.

3.3.3 Development of the Predictor Model

Introduction to Time-Series Modeling

To develop a prediction model based on time-series input data, deep learning methods such as Recurrent Neural Networks (RNN) and Long-Short Term Memory (LSTM) neural networks have been widely explored and applied across various fields, including chemical engineering [70]. These neural networks are capable of processing long sequences of data and learning about the information between each element. This results in the models learning about the correlations within the data sequence. This capability enables RNNs and LSTMs to better understand and fit

time-series data compared to traditional feedforward neural networks (FNN).

A novel deep learning model, the transformer, has emerged in recent years and rapidly gained significant attention in the field of natural language processing (NLP) due to its outstanding and record-breaking performance on most NLP tasks [46]. The transformer network structure has also demonstrated equally impressive performance in computer vision tasks such as image classification and object detection [47]. Additionally, one of the most popular and advanced examples of Artificial Intelligence (AI), the large language model (LLM), which includes products such as ChatGPT and BERT, employs the transformer architecture. By utilizing billions of training data points and model parameters, LLMs are capable of generating human-like responses and solving a wide range of problems, offering substantial potential for future applications [110].

The transformer model employs an encoder-decoder architecture to handle a sequence of time-series data. A multi-head self-attention mechanism [86] is used within each block to compute the relevance of each element in the sequence relative to every other element. This approach enables the model to effectively capture intercorrelations across the entire sequence of time-series data. The attention mechanism calculates attention scores for each pair of elements, which are then normalized using a softmax function to aggregate the attention values. This process allows the transformer to maintain a comprehensive understanding of the relationships within the data sequence, surpassing the capabilities of traditional LSTM networks [16, 86].

LSTM networks, while effective in handling time-series data and addressing the vanishing gradient problem occur in simpler recurrent neural networks (RNNs), can still suffer from memory loss over long data sequences. This limitation arises because LSTMs process data sequentially, which struggles to capture long-range dependencies. In contrast, the self-attention mechanism of the transformer allows it to simultaneously consider all positions in the sequence, ensuring that long-term dependencies are preserved and effectively learned. This ability to maintain a global perspective on the data sequence makes the transformer particularly well-suited for tasks that require a deep understanding of a long sequence of time-series data.

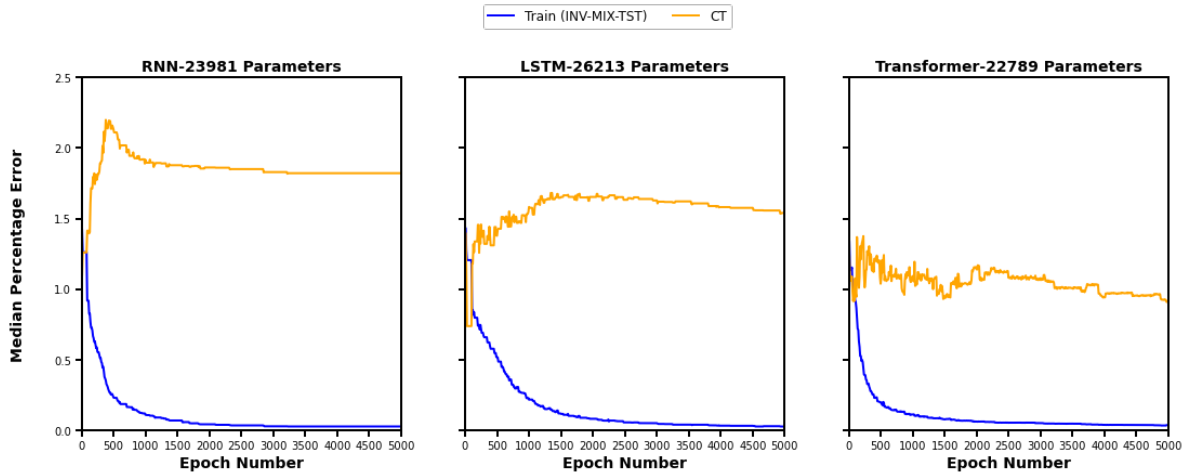


Figure 3.5: Comparison between RNN (left), LSTM (center), and Transformer (right) models. Two datasets of real-time process data were considered: a mixed dataset from multiple processes that was used for both training and validation (blue), and a dataset from a single process that was only used for validation (orange).

In Fig. 3.5, a Transformer, Recurrent Neural Network (RNN), and Long-Short Term Memory (LSTM) model are compared to demonstrate their respective effectiveness at training models. The number of optimized parameters for the three tested models are intentionally selected to have similar values for the purpose of comparison. Both the RNN and LSTM networks have one recurrent layer with 24 neurons, and the output vectors of each recurrent unit are concatenated together and fed to a FNN network with one hidden layer that contains 32 neurons to generate the output vector. Fig. 3.5 shows that the Transformer model outperforms the other two models in terms of both learning efficiency on the blue training dataset and median percentage error on the orange unseen test dataset, which has a different disturbance from the training dataset. Thus, the ability of the transformer model to maintain a general, global perspective on the data sequence makes it particularly well-suited for tasks that require a deep understanding of a long sequence of time-series data.

Transformer Soft Sensor Structure

The overall structure of the transformer soft sensor for the ALE process is shown in Fig. 3.6. Following the input layer, a dense layer, or embedding layer, is used to linearly encode the input

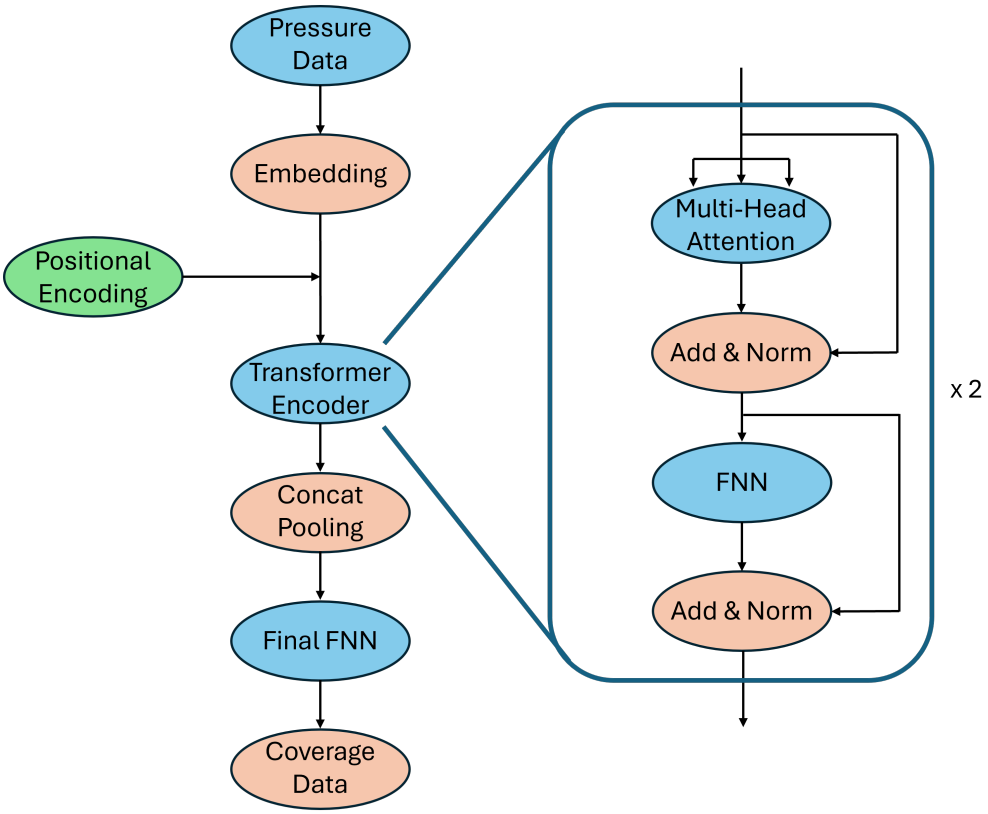


Figure 3.6: Structure of the Soft Sensor Transformer Network. Pressure data is embedded by a dense layer with dimension 8, undergoes positional encoding, and then is fed into 2 identical multi-head encoder blocks. Inside the encoder block, there is an internal FNN with a hidden layer of 64 neurons. The output of the encoders are combined by concatenation pooling and then fed into the final FNN, which has 2 layers of 64 neurons each, and it outputs the reaction coverage.

data into a suitable format for the transformer encoder blocks. The applied transformer network consists solely of encoder blocks, as the task only involves regression and does not require the model to generate a series of future predictions.

Positional encoding is added to the input sequence of the embedded vector of wafer surface pressure time-series data because the transformer blocks do not inherently recognize the order of the input elements. This encoding layer provides the necessary information regarding the order of the input elements to the transformer encoder blocks. The positional encoding equations have the following form:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{3.8}$$

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{3.9}$$

where PE is the positional encoding value, i is the index indicator within a single vector in the sequence that defines the pressure measurement vector at a specific time, $2i$ represents an even index position in the vector, $2i + 1$ corresponds to an odd index position, pos is the index number of the vector in the sequence, and d_{model} is the embedded vector dimension.

The encoder block employs the principle of residual learning and includes a single-layer neural network to process the output data from the transformer encoder blocks. Subsequently, a concatenation pooling operation is applied, which connects and aggregates the outputs from all the transformer blocks from each element in the sequence to prevent information loss. This pooled data is then fed into a final FNN to produce the total wafer coverage values. The hyperparameters of the soft sensor transformer model are summarized in Table 3.2.

Table 3.2: Hyperparameters for the transformer model.

Model Hyperparameter	Value
Input Dimension	5
Embedding Dimension	8
Number of Heads	2
FNN Neurons	64
Dropout Ratio	0.25
Encoder Layer Number	2
Final FNN Layer Number	2
Final FNN Neurons	64
Output Dimension	5

Transformer Model Training

The data is separated into two parts: 80% of the data is used for training, while the remaining 20% is used for validation testing. The training-testing split is conducted randomly using the dataset splitting module from the scikit-learn package. The Mean Squared Error (MSE) loss function is applied as the loss criterion, and the ADAM optimizer is employed for model parameter updates. The loss value on the validation dataset is recorded at each training epoch, and the model with the lowest MSE on the validation dataset is saved as the most up-to-date candidate model. The training hyperparameters, including batch size, number of training epochs, learning rate, and initialization algorithm, are all optimized using a trial-and-error method based on the model performance on the validation dataset. This approach is feasible and time efficient both because the dataset size is not large and because the transformer architecture is well suited for parallel computing and training. The final optimized hyperparameters for the model included a batch size of 64, 700 total training epochs, a learning rate of 0.001, and the Xavier initialization method. The random seeds used in data splitting and model training were also fixed to improve reproducibility.

3.4 Heuristic Analysis Method

One method to guarantee creating the most optimal model is to simply create models for every possible combination of process-specified datasets. By doing so, and then comparing their Mean Squared Errors (MSE), it is trivial to pick out the model with the best performance. However, this process becomes increasingly inefficient as the number of possible datasets increase. Specifically, there will be $n!$ possible models for n datasets. And in industry, where there are often hundreds of datasets for a particular etch process, it is unrealistic to exhaustively search through every single dataset combination. Thus, there is a motivation to create a heuristic that determines which datasets should be aggregated.

3.4.1 Statistical Methods Introduction

A well-designed heuristic is essentially a distillation of the intrinsic statistical characteristics of a dataset. One way to extract this statistical information is to train models on exactly one process-specified dataset and then compare their performances on the overall dataset that is the combination of all process-specified datasets. This method reduces the number of models that must be trained to determine the most optimal model, changing it to scale linearly with the number of datasets.

The ideal datasets to aggregate are ones that are different from one another. Training the model on a wide variety of data will allow it to better understand and predict edge-case scenarios, which are oftentimes when the run fails. This selection criteria for dataset aggregation can be translated into selection criteria for the models trained on single process-specified datasets: aggregate datasets whose representative models are complementary to each other. If one model performs sufficiently on Dataset B and poorly on Dataset C, then the ideal dataset to aggregate it with would be one whose representative model performs well on Dataset C and poorly on Dataset B.

Two statistical methods of capturing this relationship are explored in this chapter: covariance and the Pearson Correlation Coefficient (PCC). The covariance measures the general strength of

the relationship between any 2 datasets, and the PCC measures the level of linear correlation between any 2 datasets. Thus, both statistical methods will be used to create heuristics that evaluate which datasets should be aggregated. Then, by comparing those results to that of the brute-force exhaustive search, the better heuristic will be determined.

3.4.2 Heuristic Evaluation Method

First, a model is trained on each process-specified dataset as specified in the "Process Datasets" section. Then, these single-process models are run on an aggregate dataset consisting of all 4 process-specified datasets to create a set of 4 MSE datapoints for each single-process model. Finally, covariance and PCC is calculated between each pair of models for their MSE datapoints. Covariance is specifically calculated with the formula below:

$$Covar_{X,Y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{N-1} \tag{3.10}$$

where X, Y represent any two models, x_i is MSE of model X for dataset i, $\bar{x}$ is the average MSE of model X, y_i is MSE of model Y for dataset i, $\bar{y}$ is the average MSE of model Y, and N is the number of datasets. In this case, $N = 4$. The PCC is similarly calculated with the formula below.

$$\rho_{X,Y} = \frac{Covar_{X,Y}}{\sigma_X \sigma_Y} \tag{3.11}$$

where $Covar_{X,Y}$ is the covariance between any two models X, Y as calculated in Eq. (3.10), σ_X is the standard deviation of model X, and σ_Y is the standard deviation of model Y.

Finally, for any given model trained on aggregated datasets, we can determine the heuristic value of that model by averaging the covariance or PCC values for the composite pairs of the aggregated dataset. For example, if the aggregated dataset contains datasets X, Y, Z, then the composite pairs are every unique pair: (X, Y); (X, Y); (Y, Z). Then, the covariance heuristic

value of a model trained on datasets X, Y, Z can be found with:

$$h_c(M(X,Y,Z)) = \frac{Covar_{X,Y} + Covar_{X,Z} + Covar_{Y,Z}}{3} \tag{3.12}$$

where h_c is a function that returns the covariance heuristic value of a model, $M(X,Y,Z)$ is the model trained on datasets X, Y, Z, and $Covar_{X,Y}$ is the covariance as defined in Eq. (3.10). Similarly, the PCC heuristic value is defined as:

$$h_\rho(M(X,Y,Z)) = \frac{\rho_{X,Y} + \rho_{X,Z} + \rho_{Y,Z}}{3} \tag{3.13}$$

where $h_\rho()$ is a function that returns the PCC heuristic value of a model and $\rho_{X,Y}$ is the PCC as defined in Eq. (3.11). With Eqs. (3.12) and (3.13), we can now calculate heuristic values for any model based on the datasets used to train that model.

3.5 Predictor Model Results and Discussion

As shown in Table 3.1, there are 4 process-specified datasets. This means that there are 15 possible dataset combinations that can be used to train a model: 4 models are trained on only 1 dataset, 6 models are trained on 2 datasets, 4 models are trained on 3 datasets, and 1 model is trained on all 4 datasets. There are 2 methods to evaluate the performance of these 15 models. The first is to examine the MSE of the model when it is run on the validation portion of the process-specified datasets, and the second is to examine the accuracy of the model when run on the random-run dataset described in the "Process Datasets" section. The full results of the MSE test are shown in Table 3.3.

where the columns represent the validation dataset the model was run on, the rows represent the training datasets of the model, and the values in the table are the resulting MSE. The MSE values are colored such that low MSE scores are green and high MSE scores are red.

Table 3.3: MSE of each model for each validation dataset.

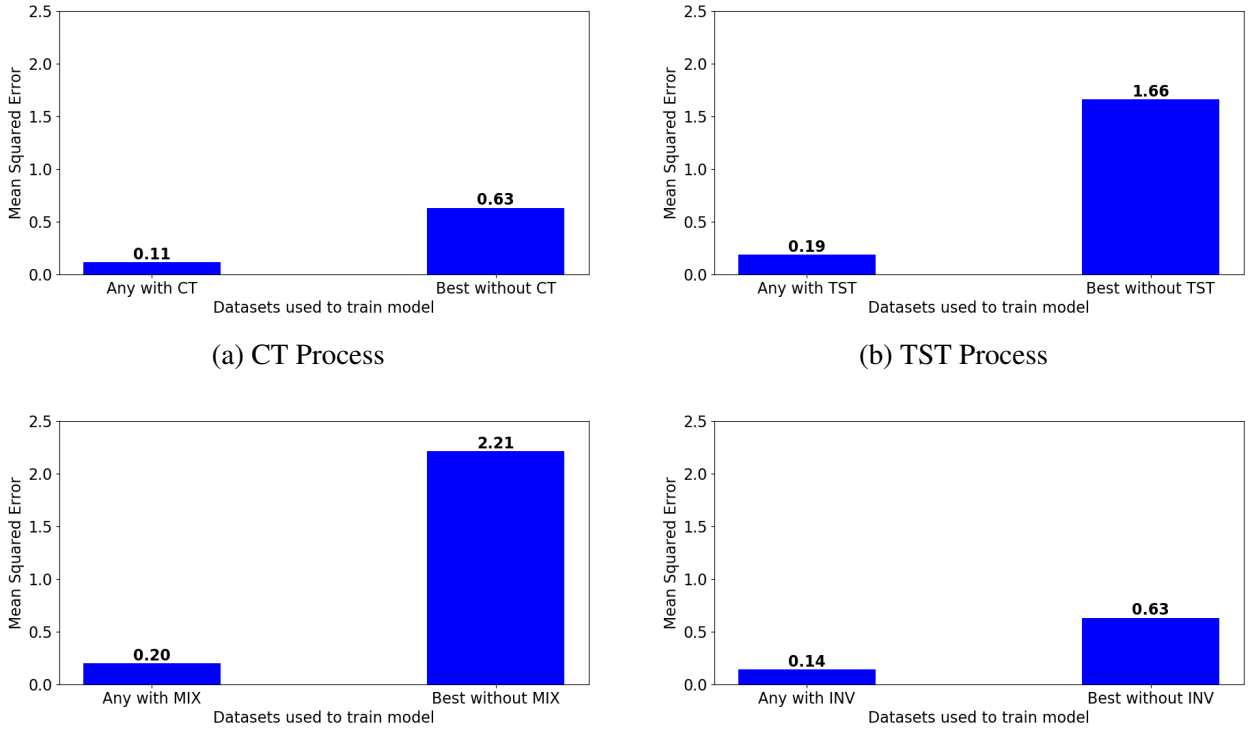
Processes Trained On	Processes Tested on				
	CT	INV	MIX	TST	All
CT	0.03	0.63	2.46	1.82	1.24
INV	0.63	0.03	2.46	1.69	1.20
MIX	2.36	2.60	0.05	0.18	1.67
TST	3.35	3.69	2.21	0.18	2.36
CT+INV	0.05	0.04	2.46	1.95	1.12
CT+MIX	0.21	0.95	0.32	1.75	0.81
CT+TST	0.08	1.02	2.31	0.13	0.89
INV+MIX	1.38	0.31	0.37	1.66	0.93
INV+TST	0.75	0.25	2.41	0.33	0.94
MIX+TST	2.16	2.41	0.24	0.24	1.26
CT+INV+MIX	0.13	0.15	0.16	1.78	0.56
CT+INV+TST	0.11	0.12	2.26	0.15	0.66
CT+MIX+TST	0.13	1.07	0.17	0.18	0.39
INV+MIX+TST	1.15	0.08	0.10	0.11	0.36
All	0.16	0.15	0.17	0.17	0.16

3.5.1 Multi-Process Model Performance

From Table 3.3, the model with the best performance across all 4 processes is the one trained on all 4, with an MSE of 0.16. However, it is difficult to grasp why just by examining the complete tabulated results. To better compare the 4 datasets, we begin by examining the degree of independence between them. In other words, we want to know if it is possible to create a strong model for a dataset without training the model on that very same dataset.

To answer this question, we compared the average performances of the models trained on a particular process to the performance of the best model not trained on the process. Figs. 3.7a to 3.7d are bar charts showing this comparison for each process.

Figs. 3.7a to 3.7d demonstrates that, for all processes, the model performance is drastically improved whenever the model is trained on the corresponding dataset of the process. This illustrates



(a) CT Process

(b) TST Process

(c) MIX Process

(d) INV Process

Figure 3.7: These subfigures measure model performance by comparing their mean squared errors when tested on the validation dataset of the stated process. The left bar is the **average** performance of all models trained on the stated process, and the right bar represents the **best** model that was **not** trained on the stated process.

that the failure mechanism for each dataset is unique. Thus, if a model is missing any one dataset, it will perform badly for the validation portion of that dataset. On the other hand, if there was a process that was not independent, then the dataset of that process can be excluded from the training dataset for the best performing model.

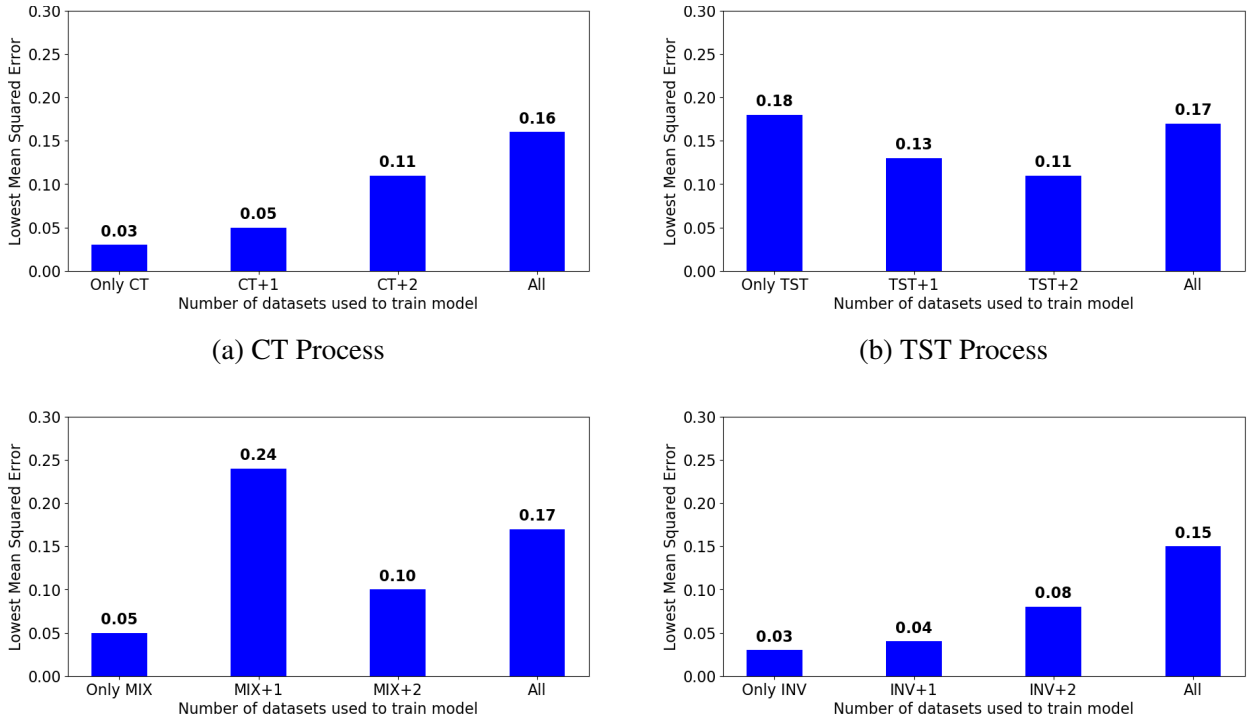
To demonstrate this point, let p_d be a process that is not independent; that is to say, there must be a model m_{nd} that is trained on the processes p_i, p_j but performs as well as a model trained on p_d. As other independent process datasets are aggregated on, any models whose training datasets include p_i, p_j and omit p_d will still perform well on the p_d. Thus, the model that performs best on the union of all the process datasets can omit p_d as long as p_i, p_j is included. By repeating this procedure, all the non-independent processes can be omitted to yield the ideal training dataset.

So, for a manufacturing environment that spans the full range of all the individual process datasets, the best predictor model will be trained on an aggregated dataset that includes the dataset of all independent processes. This environment is equivalent to one where the process runs have relatively high fail rates due to volatile processing conditions.

3.5.2 Single-Process Model Performance

It is also important to understand how to best optimize a model for performance on a single process, which is more representative of a low variance process than multiple processes. This is found by examining Table 3.3 and determining which model has the smallest MSE in each column. The best model for each process is the model trained only that process, except for TST; the best model for that process was trained on CT+TST. This demonstrates that adding the data of other processes into the training data dilutes the amount of representative process data for that process, which generally causes the model performance to decrease. To gain more insight into how aggregating data affects the model performance on a single process, Figs. 3.8a to 3.8d illustrate the MSE of the best performing model as a function of the number of datasets used to train it.

Figs. 3.8b to 3.8d demonstrate that, generally, the performance of a model on a single process



(a) CT Process

(b) TST Process

(c) MIX Process

(d) INV Process

Figure 3.8: These subfigures measure model performance by comparing their mean squared errors when tested on the validation dataset of the stated process. From left to right, the number of datasets used to train the model increases from 1 to 4 as stated in the x-axis, with the stated process dataset always being included. Each bar represents the highest performing model for that number of datasets.

decreases as data from other processes is aggregated into the training dataset. This holds true for both the CT and INV processes, but the MIX and TST processes deviate from the other 2 processes.

While the general trend for MIX as shown in Fig. 3.8c is that the single-process model performance decreases with increasing aggregation, the model trained on 2 datasets performs unexpectedly poorly. This can be attributed to the fact that MIX is the most independent process. In the previous section, MIX had the highest MSE when looking at models not trained on the process. This means that MIX cannot be represented well by an aggregation of the other process datasets. So in the case of single-process models trained on 2 datasets, MIX will be diluted the most compared to the other 3 processes, which causes it to have an outlier in terms of performance.

TST actually demonstrates the inverse of the expected response where the single-process model performance increases with increasing aggregation. This is likely due to the fact that TST had the highest MSE when looking at models trained on that process among Figs. 3.7a to 3.7d. Because even the models trained on the TST process dataset had mediocre performances on the TST validation dataset, it demonstrates that the Transformer method struggles to create a good model for this dataset. But as more datasets are aggregated into the model, the model performance improves. This demonstrates that data aggregation can also help improve model performance for processes that are difficult to model with just their own process data.

From the multi-process and single-process analyses, it is seen that the ideal amount of data aggregation depends on not just the characteristics of the process datasets, but also on the process environment. If there are multiple processes with poor controls and large variability in between runs, then the model should be trained on as many independent datasets as possible. For single process models, while there is a general pattern for how data aggregation affects model performance on single processes, there are enough exceptions that the pattern cannot be used to determine what datasets will yield the model with the best performance. Thus, to further improve manufacturing efficiency, there needs to be a method to choose which datasets to aggregate together to train a predictor model for low-variance processes.

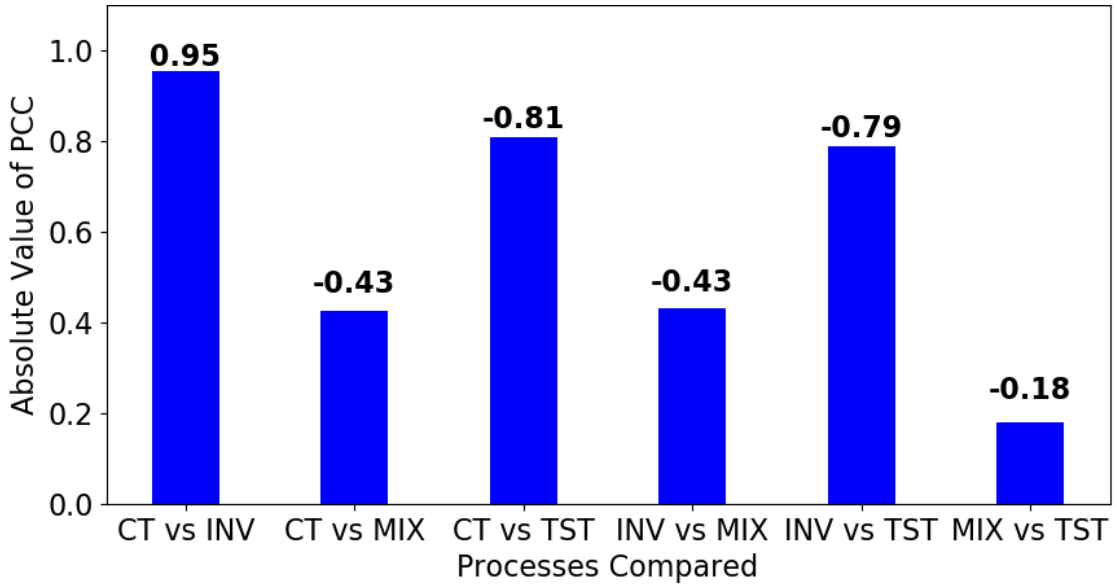


Figure 3.9: Comparison of the Pearson Correlation Coefficient (PCC) values for each pair of processes. The datasets used to calculate the PCC are the MSE values of a single-process model for each process.

3.5.3 Heuristic-based Assessment of Datasets

Another way to analyze model performance for low-variance processes is to observe their performance on the random-run dataset as described in the "Process Datasets" section. These datasets represent a process environment where runs are expected to pass and where most runs have similar kinetic parameters. Additionally, we will explore the heuristics described in the "Heuristic Analysis Method" section to see if they can predict which datasets should be aggregated for a low-variance process environment. The 2 proposed heuristics of covariance and PCC are examined, and the values of the statistical methods for each pair of single-process models are shown below.

Note that the sign of the heuristic represents whether the relationship between the 2 datasets is positively or negatively correlated. This relationship is not important, as we are only concerned with how strong the correlation between 2 datasets is, not the direction of said correlation. Thus, we are only interested in the magnitude of the calculated covariance and PCC. So, for each dataset, we find the covariance and PCC between it and all the other datasets, which is shown in Tables 3.4

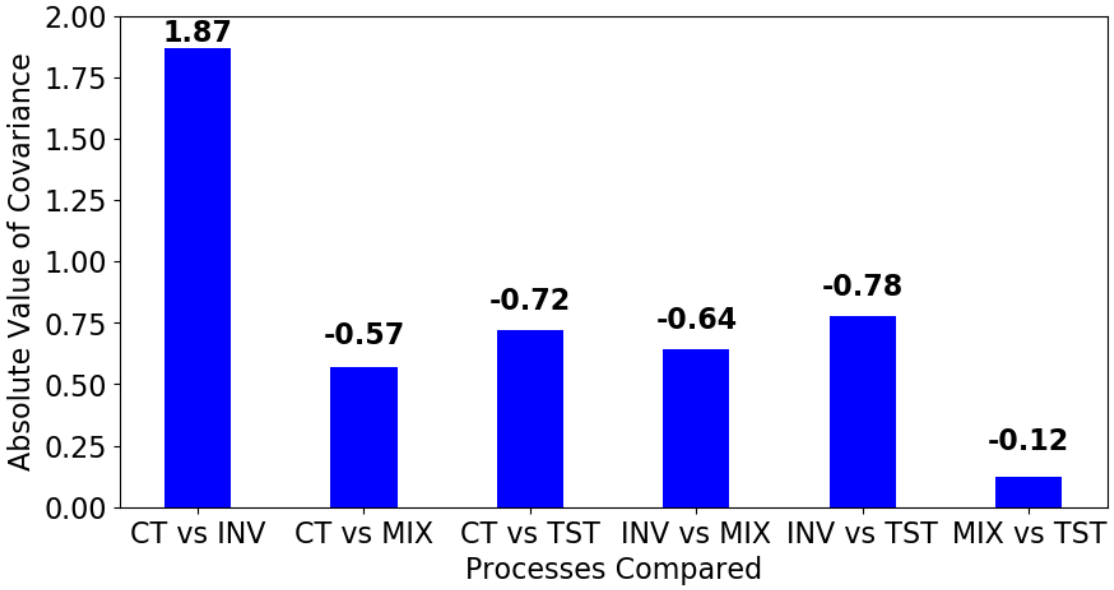


Figure 3.10: Comparison of the covariance values for each pair of processes. The covariance is calculated with the same datasets used to calculate the PCC.

and 3.5. Each entry is colored such that lower values are green and higher values are red.

Table 3.4: Average absolute covariance values for each dataset.

	CT	INV	MIX	TST
CT	1.00	1.87	0.57	0.72
INV	1.87	1.00	0.64	0.78
MIX	0.57	0.64	1.00	0.12
TST	0.72	0.78	0.12	1.00

Table 3.5: Average absolute PCC values for each dataset.

	CT	INV	MIX	TST
CT	1.00	0.95	0.43	0.81
INV	0.95	1.00	0.43	0.79
MIX	0.43	0.43	1.00	0.18
TST	0.81	0.79	0.18	1.00

With Tables 3.4 and 3.5, we can predict what the covariance and PCC will be for a model trained on multiple datasets by following the procedure described in the “Heuristic Evaluation

Method" section. Note that it is desired for the predicted heuristic value to be low. The lower the predicted value, the less overlap there is between each constituent dataset, making that combination of datasets more likely to better represent the overall process.

To test the performance of the two proposed heuristics, we created and ran four models on the random-run dataset described earlier. Then, we evaluated the covariance and PCC heuristic for each model to see how well they predict the ranking of the models. The results are summarized in Table 3.6. The first 2 columns represent the true prediction ability of the model, the 3rd represents the true classification ability of the model, the 4th represents the predicted performance by the PCC heuristic, and the 5th represents the predicted performance by the covariance heuristic.

Table 3.6: Ranking of 4 models.

		Etch MSE Median	SD% MSE Median	Acc.	Corr	CoV
Trained On	CT+INV+MIX	4.87×10^{-7}	1.95×10^{-2}	95.00%	0.603	1.027
	CT+INV+TST	5.22×10^{-7}	2.41×10^{-2}	96.67%	0.850	1.123
	CT+MIX+TST	5.30×10^{-7}	3.55×10^{-2}	96.67%	0.473	0.470
	INV+MIX+TST	4.95×10^{-7}	2.42×10^{-2}	98.33%	0.467	0.513
	All	1.78×10^{-6}	4.15×10^{-2}	93.33%	0.598	0.783

The true performance of each model shown in Table 3.6 is represented in the Acc. column. From it, we can see that the model trained on the INV, MIX, and TST process datasets performed the best. The PCC heuristic predicts this correctly, assigning it the lowest score. However, the covariance heuristic failed to do so, instead predicting that the model trained on the CT, MIX, and TST process datasets would perform the best. Another point of comparison is to see which heuristic makes the most correct predictions. The PCC heuristic correctly predicts that CT+INV+MIX is ranked 4th, incorrectly predicts that CT+INV+TST is ranked 5th, correctly predicts that CT+MIX+TST is ranked 2nd, correctly predicts that INV+MIX+TST is ranked 1st, and incorrectly predicts that All is ranked 3rd. Overall, it made 3 correct predictions. On the other hand, the covariance heuristic correctly predicts that CT+INV+MIX is ranked 4th, incorrectly

predicts that CT+INV+TST is ranked 5th, incorrectly predicts that CT+MIX+TST is ranked 1st, incorrectly predicts that INV+MIX+TST is ranked 2nd, and incorrectly predicts that All is ranked 3rd for a total of only 1 correct prediction. For both metrics, the PCC heuristic outperformed the covariance heuristic. Thus, these results demonstrate that the PCC heuristic is more accurate at predicting the performance of aggregated dataset models than the covariance heuristic.

3.6 Conclusion

To supplement the growing need for increased semiconductor manufacturing efficiency, a novel real-time etch rate predictor model was created with simulated process data. A multiscale method that encompasses both the macroscopic and mesoscopic domains was used to simulate the real-time evolution of atomic layer etching of aluminum oxide. With this method, four different, unique process datasets were created with varying kinetic parameters. Then, predictor models were trained on various combinations of these datasets. It was found that, for systems with high process variance, aggregating all the datasets resulted in the best performance, but for systems with low process variance, aggregating all the datasets would not result in the best performance. Because most manufacturing environments strive for low process variance, it is thus necessary to determine a way to estimate which datasets to aggregate for low process variance environments. We proposed two possible heuristics to choose datasets to aggregate: covariance and the Pearson Correlation Coefficient (PCC). After comparing model performances on a dataset of consecutive process runs, it was found that the PCC heuristic was the best predictor of performance for models trained on aggregated data. Further research into other possible heuristics, the many applications of an accurate real-time predictor model, and the scalability of these findings to larger groups of datasets is still needed, but the initial results indicate that such models can be effectively and easily created. In another forthcoming work, we have demonstrated the approach presented in the present chapter using industrial data.

Chapter 4

Integration of On-Line Machine Learning-Based Endpoint Control and Run-to-Run Control for an Atomic Layer Etching Process

4.1 Introduction

Electronic devices are an integral part of our modern society. They are used in everything from personal computers and mobile devices to smart vehicles and medical equipment. While there are many different types of electronic devices, all of them share a key feature: they are all made of densely connected integrated circuits, which are in turn composed of semiconductor transistors. Besides just the growing demand for the raw quantity of semiconductor chips, these chips are also becoming more and more compact [76, 87]. Moore's Law continues to hold true as the semiconductor chips used in these electronic devices shrink and become more sophisticated [2]. Many new semiconductor device architectures, such as the gate-all-around structure, require the

development of novel, high-precision manufacturing processes [56].

A major factor in the industry's ability to continuously manufacture more and more sophisticated chips is the usage of advanced equipment and processes, including extreme ultraviolet (EUV) lithography [89]. Additionally, other critical process steps necessitate 3D, nanoscale precision, contributing to the ongoing semiconductor supply shortages [87]. Other processes capable of achieving this high level of precision are Atomic Layer Etching (ALE) and Atomic Layer Deposition (ALD). These two processes are similar to traditional etching and deposition processes, except they use half-reactions to etch or deposit a single atomic layer of material at a time [22, 34]. This is an extremely high precision process, which is required to manufacture modern semiconductor devices with tight process specifications [2, 75, 85]. For the purposes of this chapter, the ALE of bulk Al_2O_3 is considered. This process consists of two steps: first, a precursor of hydrogen fluoride (HF) is used to prepare the Al_2O_3 surface by fluorinating it. Then, an etching reagent of trimethylaluminum (TMA) is used to etch the fluorinated surface [23, 42]. This cycle removes a single layer of the Al_2O_3 substrate and can be repeated to remove multiple layers with a high level of precision.

Besides advanced manufacturing methods, advanced process control methods are also vital in the improvement of the manufacturing efficiency of semiconductor chips due to the high sensitivities of these processes. For older etch processes, such as plasma-enhanced etch, a popular, high-precision feedback control method that is still used today is endpoint (EP) detection [71, 88]. By measuring the ratio of gases in the outlet stream or the voltage of the plasma, the end time of the process can be automatically determined in real time. The advantages of EP-based process control are twofold: product quality is maximized, as an accurate EP control system ensures that the reaction is carried out to completion. Secondly, reagent wastage is minimized, as the process will terminate soon after the reaction is completed. This is also an in-situ feedback control system, which means that there are no ex-situ variables that require several iterations to tune [54]. Another popular control strategy is Run-to-Run (R2R) control, which is a widely used ex-situ control

method and has been actively studied in the context of ALE processes [103]. R2R controllers operate on a batch-to-batch basis, adjusting process parameters between batches by using the measured output values from previous runs as feedback, unlike real-time feedback controllers, which make continuous adjustments. Rather, this batch-based approach enables consistent process optimization, helping to manage variations and disturbances that may arise during production.

While EP control systems are common for standard semiconductor etch processes, they have yet to be implemented for ALE processes. Most likely, this can be attributed to the high difficulty in relating a real-time measurable parameter to the completion of the process and the fact ALE processes focus on manufacturing cutting-edge devices that make traditional endpoint detection methods difficult to implement [80]. In a previous work, the authors demonstrated that machine-learning methods can be used in conjunction with process simulations to train a transformer model that uses real-time wafer surface pressure data as an input to predict whether a wafer is fully processed as an output for the ALE process described above [91]. This chapter continues on by using this transformer as a basis for a real-time endpoint feedback controller.

To minimize the financial and time-based costs of real-world testing, multiscale computational fluid dynamics simulations are widely used to model semiconductor manufacturing processes, including plasma-enhanced chemical vapor deposition [13, 109], atomic layer deposition [63], and atomic layer etching [103]. In this chapter, a multiscale simulation approach is applied, which combines macroscopic CFD simulations of a discrete feed reactor with mesoscopic kinetic Monte Carlo simulation. This integrated method provides a detailed and accurate representation of the actual physical processes.

This chapter explores the integration of both a real-time endpoint (EP) feedback control system and an ex-situ run-to-run (R2R) controller to ensure the optimal operation of an atomic layer etching (ALE) process in an industrial manufacturing environment with process disturbances. First, Section 4.2 summarizes the ALE process and how the process is simulated. Next, Section 4.3 describes the formulation and implementation of the EP control system. Section 4.4 does

the same for the R2R control system. Finally, Sections 4.5 and 4.6 analyze how effective various combinations of EP and R2R controllers are at mitigating the effects of a kinetic process disturbance across multiple process runs.

4.2 Process Description

4.2.1 Atomic Layer Etching

Atomic layer etching (ALE) is a modern semiconductor fabrication technique that uses two alternating half-reactions to achieve atomic-level control over the etching process [34]. Crucially, both of these half-reactions are self-limiting; this means that the reaction naturally slows down as it approaches completion. For a well-designed ALE process, overprocessing will only result in wasted reagents and will not misprocess the wafer.

The specific ALE process that this chapter examines is the etching of Al_2O_3 by the following reactions:

$$Al_2O_3\ (s) + 6HF\ (g) \rightarrow 2AlF_3\ (s) + 3H_2O\ (g) \qquad \text{(Reaction A)}$$

$$2AlF_3\ (s) + 4Al(CH_3)_3\ (g) \rightarrow 6AlF(CH_3)_2\ (g) \qquad \text{(Reaction B)}$$

In Reaction A, the gaseous hydrofluoric acid (HF) precursor fluorinates the Al_2O_3 surface. Then, in Reaction B, the gaseous trimethylaluminum (TMA) precursor etches away the fluorinated ALF_3 surface created in Reaction A, releasing a gaseous byproduct of dimethylaluminum fluoride (DMAF) [42].

The two half-reactions can each be split into multiple elementary reactions [100]. Generally speaking, these elementary reactions can be sorted into one of two categories: adsorption/desorption reactions and nonadsorption reactions. The kinetic rate constant of the former is modeled by the Collision Theory equation shown in Eq. (4.1) and that of the latter is modeled by the Transition

State Theory equation shown in Eq. (4.2).

$$k_{ads}(P_a, T) = \frac{\sigma_a P_a A_{site}}{Z_a \sqrt{2\pi m_a k_B T}} \tag{4.1}$$

where a is the adsorbate (HF, TMA), σ_a is the sticking coefficient between the adsorbate and the Al_2O_3 surface, P_a is the adsorbate's partial pressure on the wafer surface, A_{site} is the surface area of a single Al_2O_3 binding site, Z is the adsorbate's coordination number, m_a is the adsorbate's atomic mass, k_B is the Boltzmann constant, and T is the surface temperature of the wafer.

$$k_{nonads}(T) = \nu \exp\left(-\frac{E_A}{RT}\right),\ \nu = \frac{k_B T}{h} \tag{4.2}$$

where ν is the pre-exponential factor, h is Planck's constant, E_A is the activation energy, and R is the ideal gas constant.

With a kinetic rate constant equation for each elementary reaction, the overall reaction progression can be simulated with a kinetic Monte Carlo (kMC) algorithm. The algorithm takes place in a 300×300 grid that represents a larger reaction zone. Each point on the grid represents a reaction site, and the algorithm evaluates how the 90,000 reaction sites progress through time. The specifics of the kMC simulation method can be found in an earlier work by the authors, but a brief summary of the algorithm's implementation is given below [92].

1. Randomly select a reaction site
2. Calculate $k_{tot} = \sum_{i-1}^{n} k_i$
3. Find j such that $\sum_{i=1}^{j-1} k_i \leq \gamma_1 k_{tot} \leq \sum_{i=1}^{j} k_i$
4. Calculate $\delta t_k = \dfrac{-\ln \gamma_2}{k_{tot} A}$

where n is the number of possible reactions for the selected reaction site, k_i is a reaction rate constant for a possible reaction, k_{tot} is a site-specific constant, j is the selected reaction, γ_1, γ_2

are randomly generated numbers that are evenly distributed within the range $(0, 1]$, A is the total number of active sites, and δt_k is the time elapsed for that reaction. Another way to interpret Steps 2 and 3 is that they are randomly selecting a reaction for the site selected in Step 1, with the probability of each possible reaction being weighted by the reaction rate constant of that reaction. And Step 4 is calculating how long the reaction takes to occur within the context of the entire grid, with $\delta t_k \sim$1e-6 s. Thus, as the kMC algorithm is repeated, it will simulate the surface reaction progression at a very fine resolution.

4.2.2 Discrete Feed Reactor

The ALE process takes place inside a Discrete Feed Reactor (DFR), pictured in Fig. 4.1. This reactor operates at a constant temperature and pressure, and it allows reagents to continually flow into the reactor and byproducts to be flushed from the reactor. These characteristics all help ensure precise control over the process, which enables the process control techniques discussed later in this chapter. The precursor for the reaction, along with a carrier gas of N_2, enters the reactor from the inlet at the top. These gases are dispersed by a showerhead plate, which improves the precursor distribution on the wafer at the bottom of the reactor. Finally, any unused precursor and byproducts evacuate the chamber through the outlet at the sides [92].

The reactor is simulated through a Computational Fluid Dynamics (CFD) software called ANSYS Fluent, which calculates the unsteady-state pressure evolutions within the reactor. All the boundaries of the reactor are simulated as inert walls, except for three areas that are shown in Fig. 4.1: the inlet has a set mass flowrate as its boundary condition, the outlet has a set negative pressure differential as its boundary condition, and the wafer surface is modeled as a reaction zone where the mass source fluxes are calculated by the mesoscopic kMC simulation described in Section 4.2.1. Of note, the wafer surface is separated into 5 sections as shown in Fig. 4.2; this allows the collected process data to contain information regarding the reaction progression across the wafer itself, which is a crucial process metric. Given these boundary conditions, the fluid vol-

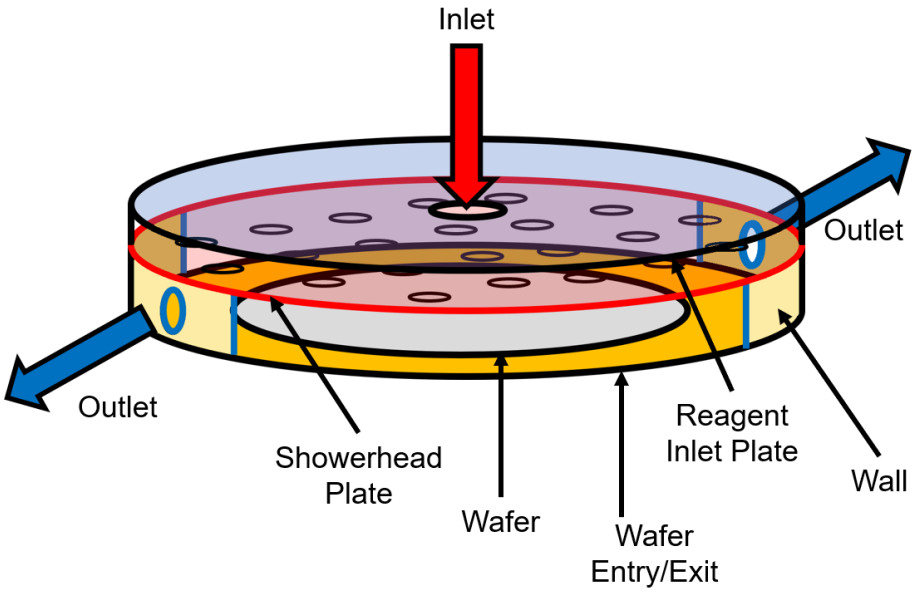


Figure 4.1: 3D representation of the discrete feed reactor and its components.

ume of the reactor is then divided into a mesh so that the characteristic mass, momentum, and energy transport equations shown in Eqs. (4.3) to (4.5) can be solved numerically.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \overrightarrow{v}) = S_m \tag{4.3}$$

$$\frac{\partial}{\partial t}(\rho \overrightarrow{v}) + \nabla \cdot (\rho \overrightarrow{v}\,\overrightarrow{v}) = -\nabla P + \nabla \cdot (\overline{\overline{\tau}}) + \rho \overrightarrow{g} + \overrightarrow{F} \tag{4.4}$$

$$\frac{\partial}{\partial t}(\rho E) + \nabla \left(\overrightarrow{v} \left(\rho E + P \right) \right) = -\nabla \left(\Sigma h_j \overrightarrow{J}_j \right) + S_h \tag{4.5}$$

where ρ is the gas-phase species density, $\overrightarrow{v}$ is the velocity of said species, S_m is the source generation and consumption flux of that species, P is the operating pressure of the reactor, $\overline{\overline{\tau}}$ is the normal two-rank stress tensor, $\overrightarrow{g}$ is the gravitational acceleration constant, $\overrightarrow{F}$ is the force acting on the system, E is the accumulated rate of system energy, S_h is the energy source generation or consumption, h_j is the sensible enthalpy flux of gas species j, and $\overrightarrow{J}_j$ is the mass diffusion flux of gas species j.

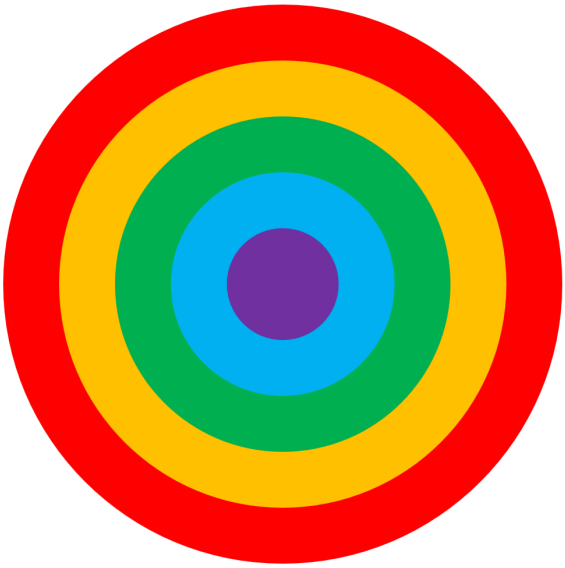


Figure 4.2: Top-down view of the various reaction zones considered in the overall simulation.

The simulation is run with an integration timestep of 0.001 s, which is how often Eqs. (4.3) to (4.5) are solved. Additionally, it was found that the operating conditions listed in Table 4.1 result in both half-reactions being fully processed within 2 s [100].

Table 4.1: Operating conditions used for all the ALE simulations in this chapter.

Variable	**Value**
Operating Temperature	573 K
Operating Pressure	300 Pa
Inlet Mass Flowrate	1e-5 kg/s
Outlet Pressure	-200 Pa
Reaction A HF Inlet Mole Fraction	0.1
Reaction B TMA Inlet Mole Fraction	0.5

4.2.3 Multiscale Model

Due to the self-limiting nature of the ALE half-reactions, the reaction rate is not constant; it gradually slows down as the reaction approaches completion. Thus, the mesoscopic kMC and macroscopic CFD simulations cannot be run independently as the former affects the latter, and vice

versa. To link the two simulations and increase the accuracy of the overall simulation, a multiscale framework is used.

The multiscale coupling method that conjoins the CFD and kMC simulations is shown in Fig. 4.3. The simulation starts by loading a a steady-state simulation of the CFD model where the input is pure N_2; as there is no reagent, the kMC simulation is inactive. Then, the unsteady-state CFD model with the inlet parameters listed in Table 4.1 is ran for a single timestep of 0.001 s. Once the CFD simulation converges, the partial pressures and temperature at each wafer section are sent to their own kMC simulation, which makes for 5 independent kMC simulations. Each simulation then calculates the reaction rate constants and extents of reaction for the next 0.001 s that the CFD simulation is about to simulate. These extents of reaction are converted into mass source generation and consumption fluxes, S_m, and used in the CFD simulation of the following timestep. This constant flow of information persists throughout the entire multiscale simulation to ensure both an accurate macroscopic CFD simulation and an accurate mesoscopic kMC simulation.

Each simulation represents a full process of a half-reaction, producing two important time-series datasets: the absolute pressure on the wafer surface, and the reaction completion percentage, or coverage, on the wafer surface. Both datasets consist of 5 data points at each timestep due to the 5 reaction zones, and an example of the former is shown in Fig. 4.4b. However, the latter is processed to yield the coverage mean and the coverage standard deviation, as seen in Fig. 4.4a, because these two metrics directly determine if a wafer was successfully processed or not.

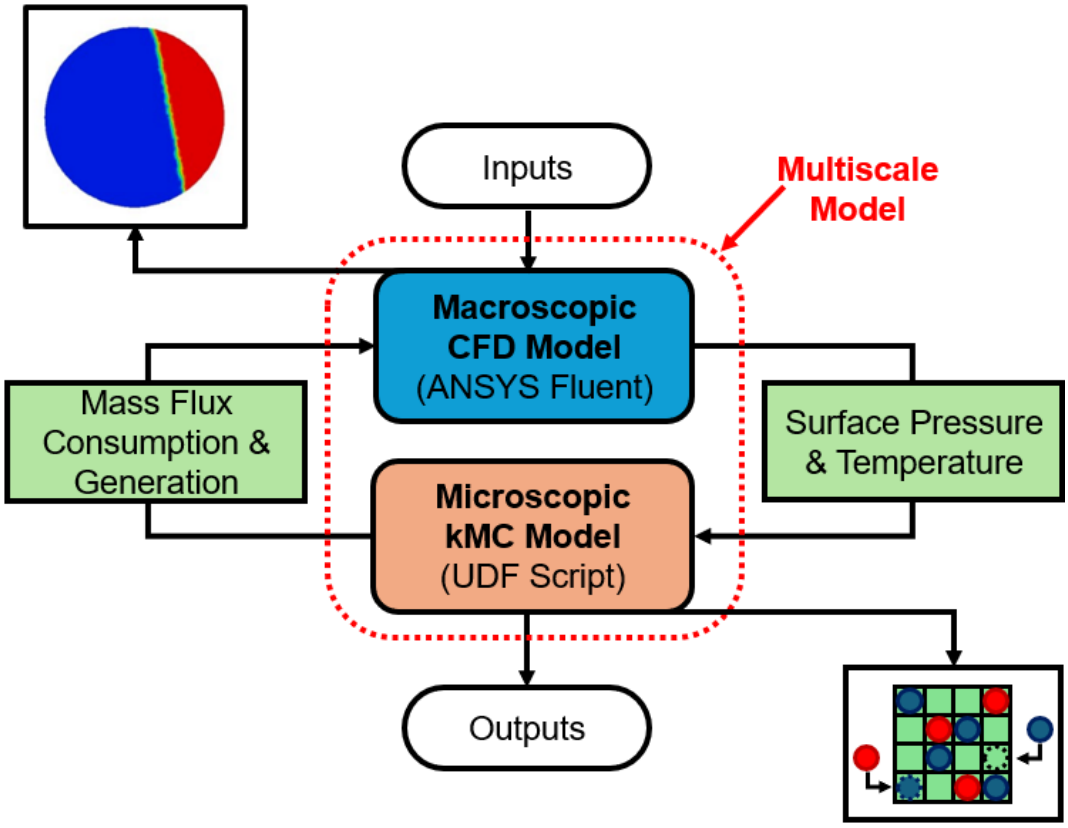
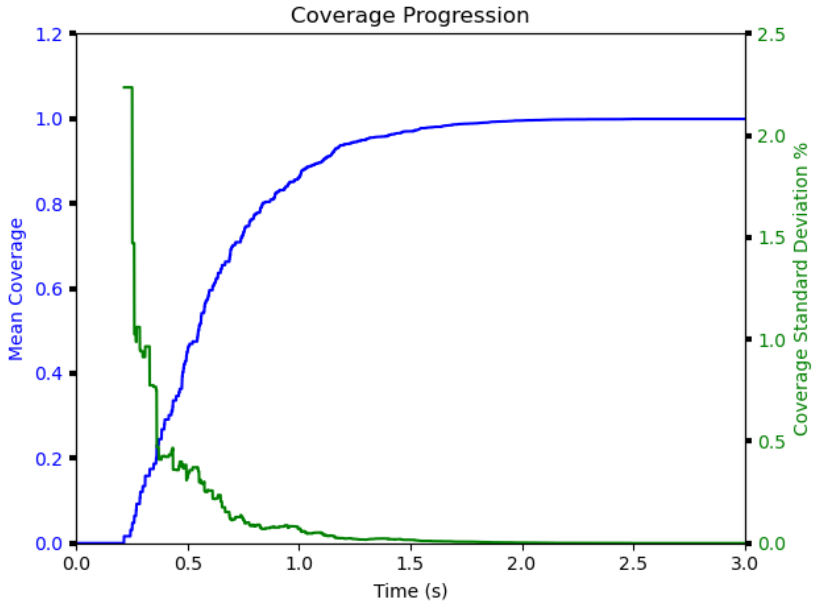


Figure 4.3: Graphical representation of the information flow in the multiscale model.

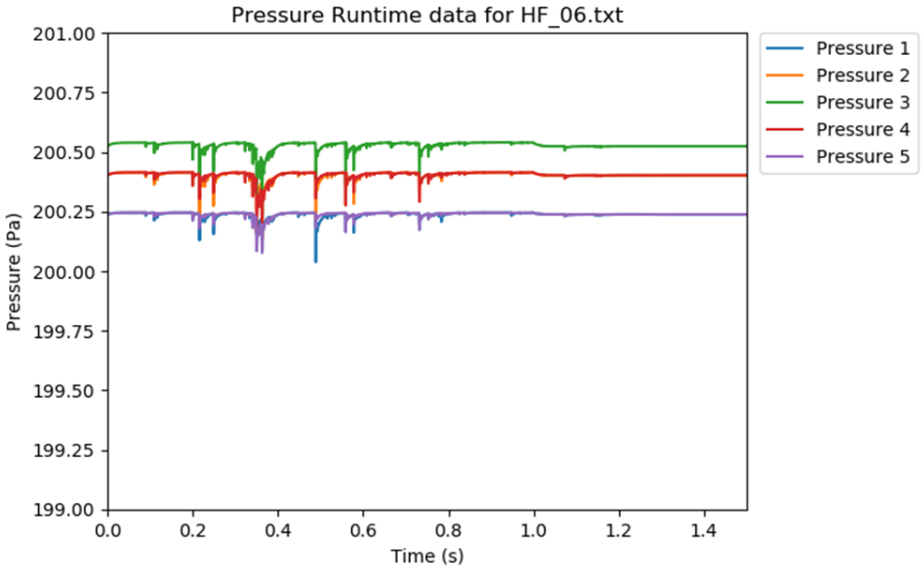
4.3 Endpoint Controller Methods

4.3.1 Endpoint Controller Description

Due to the sensitive nature of bleeding edge manufacturing techniques, process control methods are inherent assumptions. For example, proportional-integral (PI) controllers are commonly used to control the temperature and pressure of etching and deposition reactors [61]. While PI controllers work well for on-line measurable process variables to drive them to the requested setpoints, they are not suitable when the primary control objective is to control a key process parameter that cannot be measured in real-time. For example, the reaction coverage is one such parameter that can only be measured once the processing step is complete. Rather, a common process control method for well-characterized processes such as plasma-enhanced etch is endpoint (EP) control [71]. An



(a)



(b)

Figure 4.4: Example of the coverage progression data (a) and the wafer surface pressure progression data (b).

EP controller uses some sort of signal, such as a voltage change, as a flag to end the desired process. However, for the ALE process examined in this chapter, there is no such indicator. Rather, this chapter uses a data-driven transformer model to act as the indicator common in other EP controllers. Specifically, real-time pressure data is fed to the transformer model, which uses it to predict whether the process is complete. If the process is complete, then a termination signal is sent, and the process ends; if the process is not complete, then no signal is sent, and the process continues. This control system is entirely dependent on the accuracy of the transformer model and its predictions. Thus, it is vital that the transformer model is properly optimized and implemented.

The real-time endpoint detector developed in this chapter is based on a binary classifier model that uses real-time pressure data to determine whether the given ALE half-reaction has reached completion. If so, the controller ends the reaction and initiates chamber purging; otherwise, the process continues. The endpoint controller is activated 0.5 s into the process, as it is impossible for the reaction to finish any earlier than this. Once the termination signal is received, the endpoint controller is also deactivated as the reaction cannot be restarted. An optimal detector will ensure full wafer processing without wasting any precursors or time, enhancing cost-effectiveness and efficiency. However, creating such a detector is challenging, as its performance depends on accurately correlating inputs to outputs. Due to inherent limitations of the process data, it may not achieve perfect control under all conditions, leading to trade-offs discussed in later sections.

4.3.2 Transformer Model Description

To train the transformer model for the classification task described above, a total of 240 process runs with unique run conditions were simulated. Each dataset can be defined by a naming scheme with three components: Reaction, Variable, and Number. There are two possible choices for Reaction: Reaction A or Reaction B. Variable describes the general category of process condition the run was simulated under. For example, “TST” affects the pre-exponential factor, or ν, of Eq. (4.2); “CT” refers to the sticking coefficient, or σ, of Eq. (4.1); “INV” has σ and ν in an

inverse relationship; "PRESS" refers to the operating pressure inside the reactor. Finally, Number describes the exact process conditions according to Table 4.2. Each range consists of 40 points, which evenly span the range of each process condition denoted in the columns. σ represents the sticking coefficient found in Eq. (4.1), ν represents the pre-exponential factor in Eq. (4.2), and P is the operating pressure. When taken all together, "A_CT_23" refers to a process run for Reaction A with run parameters of $\sigma = 1.05,\ \nu = 1.0,\ P = 300$.

Table 4.2: List of the ranges for each variable.

Variable	σ **Range**	ν **Range**	P **Range**
TST	[1.0,1.0]	[0.5,1.475]	[300,300]
MIX	[0.5,1.475]	[0.5,1.475]	[300,300]
INV	[1.5,0.525]	[0.5,1.475]	[300,300]
PRESS	[1.0,1.0]	[1.0,1.0]	[200,395]

Finally, each process run is broken down into smaller input sequences, which is simply the pressure dataset fed to the transformer model. These input sequences represent the variable-length nature of the actual data that the EP controller is to be used on. Because each process run is simulated for 3.0 s, multiple input sequences can be extracted per process run for the purpose of model training. Specifically, the entire 3.0 s run can be separated into 26 input sequences of varying length that evenly span from 0.5 s to 3.0 s. Note that the first input sequence considered is for $t = 0.5$ s because it is considered impossible for either reaction to reach full coverage before then. Each of these variable-length input sequences have an associated output of whether the run is complete or not. In this manner, two transformer models are trained, one for each reaction, each on 1920 input sequences.

The ultimate goal of the EP controller is to take in input sequences of variable-length, time-series data and output a binary classification of whether the process is complete. There are a variety of data-driven machine learning model architectures that can handle this task, such as recurrent neural networks (RNNs), long short term memory (LSTM) networks, and transformers. In

a previous work, it was found that transformers perform the best when handling time-series input data [91]. While both RNNs and LSTM networks can handle time-series data through proper padding and masking operations, they still have their own challenges. RNNs cannot contextualize data across a long time period, "forgeting" about earlier data. On the other hand, LSTM networks can only process data in a sequential manner, making it impossible to form any long-term correlations. In comparison, the transformer's encoder/decoder structure allows it to retain information across long time periods and extract complex relationships. Transformer networks are also trained faster, as they have a parallel structure that naturally lends itself towards graphical processing units (GPUs). As RNNs and LSTM networks have sequential structures, they cannot take advantage of a GPU's powerful processing capabilities. Thus, the EP controller's process model is based on a data-driven transformer.

4.3.3 Transformer Model Training

Before discussing the transformer architecture, it is important to understand the data used to train it. The input data sequence is the variable-length, time-series pressure profile described in Section 4.2.3, and the output data is whether that pressure profile would result in a completed reaction. While the output data is stable, the raw input data has large, sudden spikes that are a natural result of the numerical solving process, as seen in Fig. 4.5. These spikes occur because the numerical solution method is trying to minimize the overall error of the entire reactor's pressure profile, not just that of the wafer surface pressure. While a transformer model can still be trained on such data, it is obviously nonideal as the noise will reduce the model's predictive ability.

To clean the wafer pressure input data, two steps are taken. First, all outliers are dropped. For this problem, the pressure is generally confined to 200 ± 1 Pa; thus, all pressure spikes/drops of more than 1 Pa were labeled as outliers and dropped from the data. As an example, after this step is taken, the raw input data shown in Fig. 4.5 becomes the data shown in Fig. 4.6a. Second, the data is further smoothed by applying a rolling average. A window of 3 data points was used to

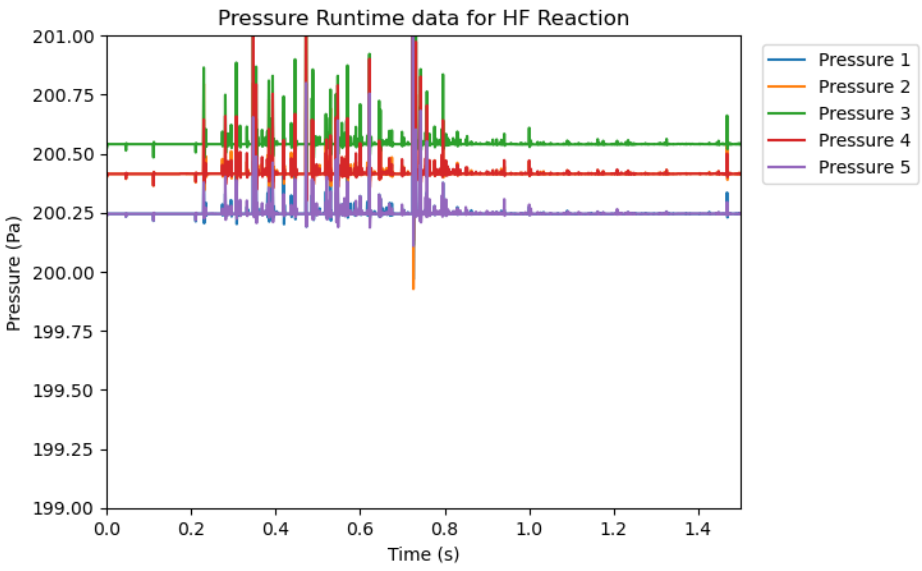
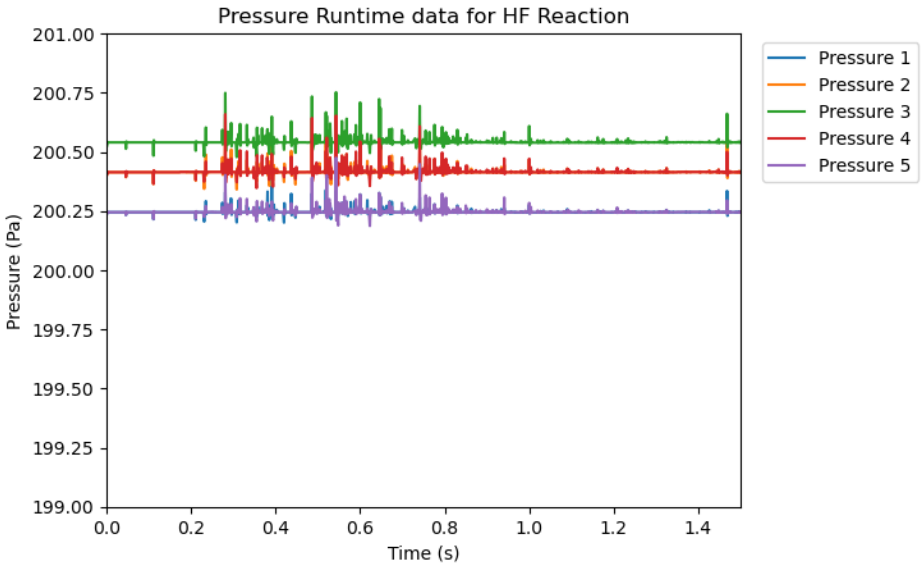


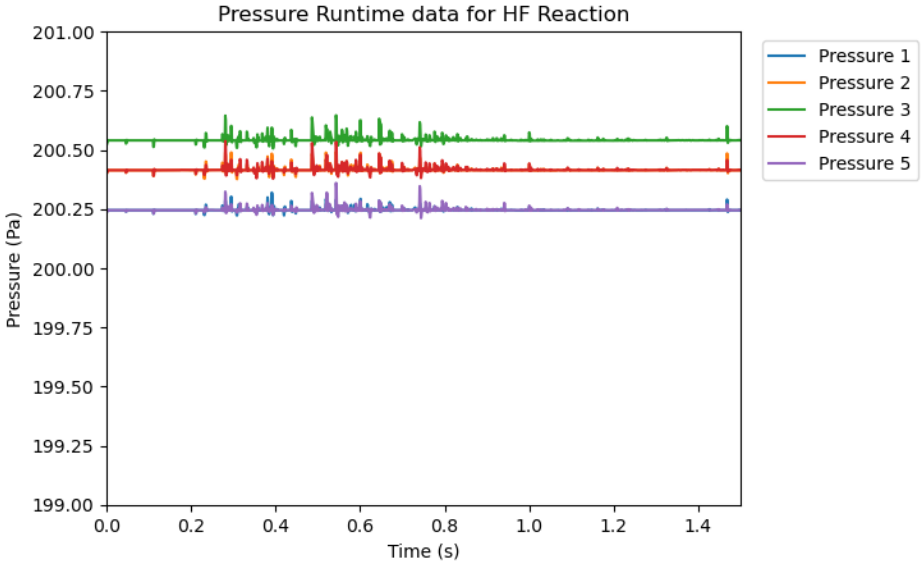
Figure 4.5: Example of the raw wafer surface pressure input data. The large pressure spikes and sharp changes make it nonideal for training a transformer model

avoid removing critical information, and the results can be seen in Fig. 4.6b. Once the input data sequence is successfully cleaned, it can be used to train and test the transformer model.

The EP controller examined in this chapter uses a transformer model, and its encoder-decoder architecture is used to correlate the wafer surface pressure to process completion. The overall structure of the transformer is shown in Fig. 4.7. Specifically, each block has a multi-head self-attention mechanism [86] that relates each element of the input sequence data to each other element; this allows the model to capture process behavior that varies over time. The real-time pressure data is fed into the model through an input layer. Following that, it is embedded by a dense layer with dimension 8 that performs a positional encoding operation that provides information regarding how the time-series elements are ordered. Inside the encoder block, there is an internal FNN with a hidden layer of 64 neurons. Two encoder blocks with multi-head attentions are stacked together in a serial manner. The outputs of the last encoder block are combined through a global average pooling operation that makes the output vector have the same dimension regardless of the length of the input sequence. This output is then fed into the final FNN, which has 2 layers of 64 neurons



(a)



(b)

Figure 4.6: Example of the wafer surface pressure input data after points more than 0.5 Pa have been removed (a) and a rolling average of 3 points is implemented (b).

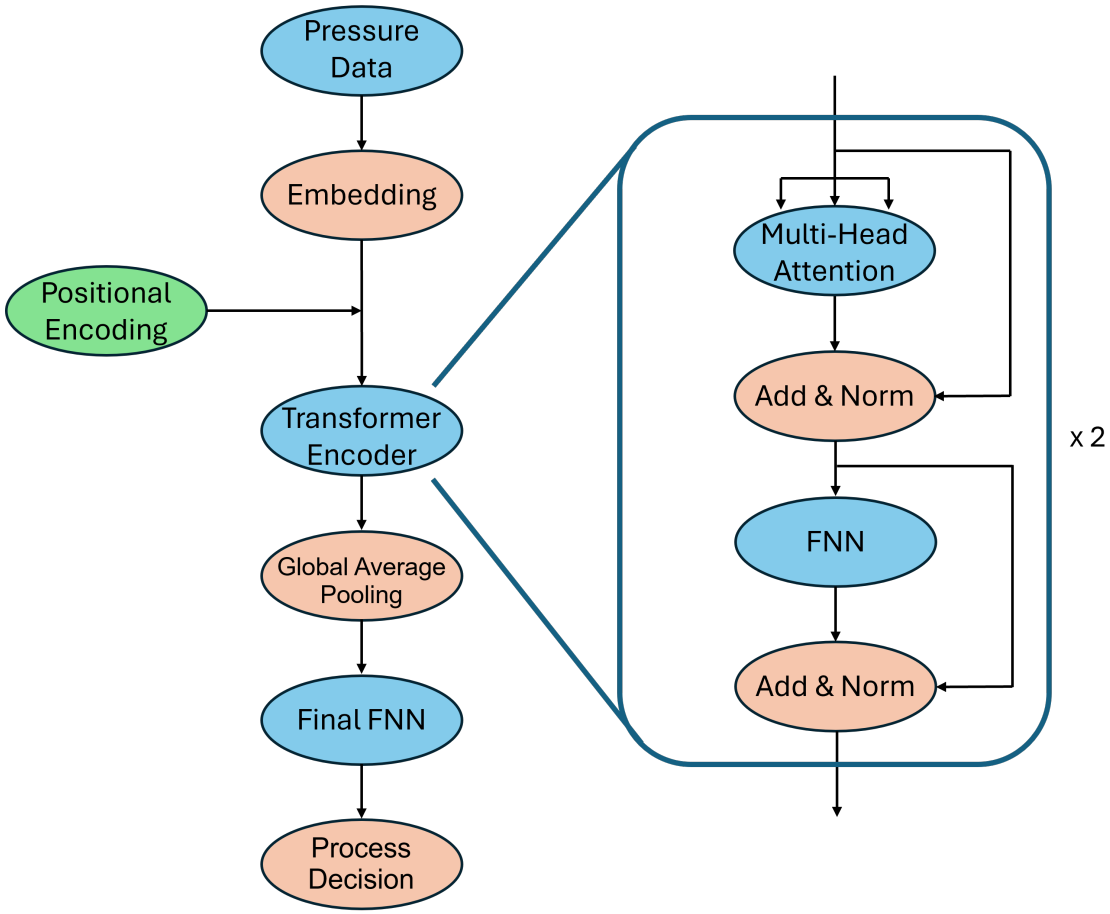


Figure 4.7: Structure of the Soft Sensor Transformer Network.

each, and it outputs the final decision with a sigmoid activation function. The hyperparameters of the soft sensor transformer model are summarized in Table 4.3.

Table 4.3: Hyperparameters for the transformer model.

Model Hyperparameter	**Value**
Input Dimension	5
Embedding Dimension	8
Number of Heads	2
FNN Neurons	64
Dropout Ratio	0.1
Encoder Layer Number	2
Final FNN Layer Number	2
Final FNN Neurons	64
Output Dimension	1

4.4 Run-to-Run Controller Methods

4.4.1 Run-to-Run Controller Description

The Run-to-Run (R2R) controller is an ex-situ controller, which means that it can only apply control actions after a process run is completed. Though it lacks real-time precision, it has access to higher-quality process data. For the ALE process, while the endpoint controller uses surface pressure as input, the R2R controller can use the final coverage as its input; this is the most important process metric, which allows the R2R controller to make finer adjustments.

Because the R2R controller only takes place after the process is completed, the final coverage that it uses as its input is actually the product of Reaction A's coverage and Reaction B's coverage; this final coverage represents the percentage of the wafer that is fully processed (underwent both Reaction A and Reaction B). While there are many parameters that the R2R controller can adjust, one of the goals of this chapter is to evaluate its efficacy when used in conjunction with the EP

controller described in Section 4.3. Thus, all the R2R controllers examined in this chapter only adjust the process time in response to the final coverage.

4.4.2 Run-to-Run Process Model

The R2R controller's control actions are generally based on a process model that describes the relationship between the process outcome and the control variable. Most such models assume a linear relationship between the outcome and the control variable, as this assumption is generally valid for small control actions. Such a linear model has the following general form:

$$y = a \cdot x + b$$

where y represents the target output, x is the control variable, a is the slope, and b is the intercept. Note that a and b are parameters that relate the behavior of the control variable to the output variable.

In this chapter, both the mean and the standard deviation (std.) of the final coverage must be controlled, as uniformity is a critical process metric in semiconductor manufacturing. Thus, there are two R2R controllers, one for each process metric. However, the calculation of the process time, which is the input, requires some nuance. Because the final coverage is a process metric that is indicative of whether *both* half-reactions were successfully completed, it is impossible for the R2R controller to independently adjust the two half-reactions' process times. Thus, the R2R controllers' process models use a process time offset term, δ, as a shared input that determines the process times for both half-reactions.

Another challenge for implementing R2R control of ALE processes is that both the mean and std. profiles exhibit highly nonlinear behavior, which poses a challenge for traditional linear control models. A poor process model can cause the model may deviate significantly from the actual physical process and result in poor control performance [104]. A solution to this issue involves

applying nonlinear transformations to the input and output parameters so that the transformed input and output have a more linear relationship. Thus, the two models will both have the form shown below.

$$\psi_c = \alpha_c \cdot \chi_c + \beta_c,\ c = m, s \tag{4.6}$$

where c is a subscript that can be either m for the final coverage mean or s for the final coverage std., ψ_c is the transformed output metric c of the process, α_c is the slope for process metric c, χ_c is the transformed input for the process metric c, and β_c is the intercept for process metric c.

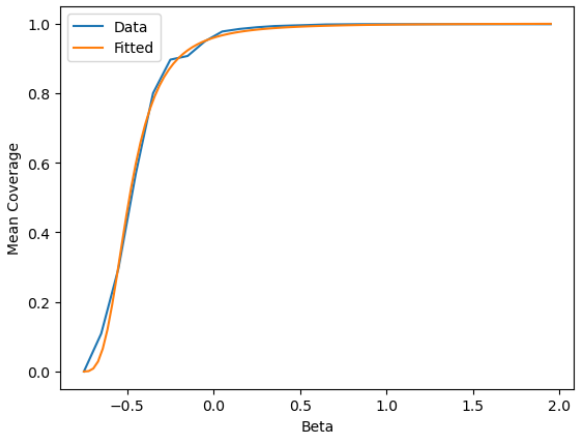
This study uses a median effect function [104] to transform the final coverage mean and a simple exponential function to transform the final coverage std. It was empirically determined that these transformed ouputs are linearly related to $\ln \delta_m$ and δ_s, respectively. The equations of the transformed terms are shown below:

$$\psi_m = \ln \frac{cov_m}{1 - cov_m},\ \chi_m = \ln(\delta_m + 0.75) \tag{4.7a}$$

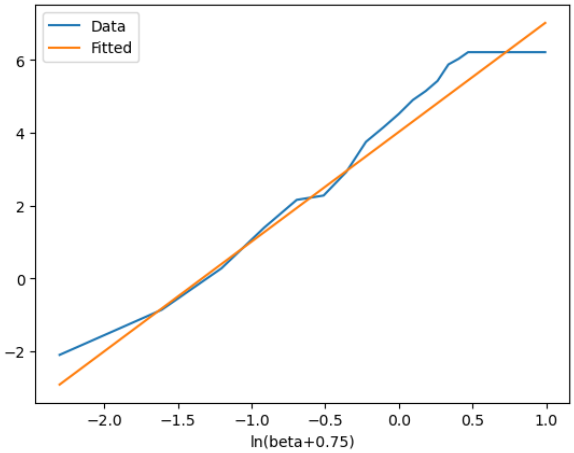
$$\psi_s = \ln cov_s,\ \chi_s = \delta_s + 0.75 \tag{4.7b}$$

where cov_m is the final coverage mean, cov_s is the final coverage std., δ_m is the process time offset for the final coverage mean equation, and δ_s is the process time offset for the final coverage std. equation. Both the nonlinear and linear fits for the final coverage mean are shown in Figs. 4.8a and 4.8b, and the fits for the final coverage std. are shown in Figs. 4.8c and 4.8d. The R^2 score of the fitting for the final coverage mean and final coverage std. are 0.997 and 0.96, respectively. These scores show that the fitted curve is highly accurate and can be used to build a process model.

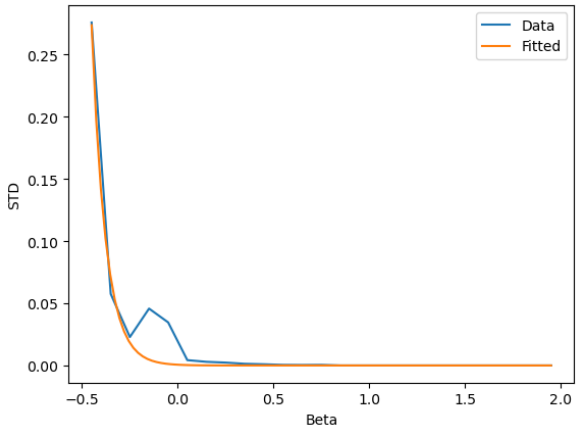
The kink in Fig. 4.8c is a result of the difference in magnitude of the kinetic rate constants. For the HF reaction, there is one surface reaction that is 1000 times larger than the rest. Thus, when the substrate reaches that step, large parts of the wafer will effectively pause at the slow reaction while the remainder finishes reacting. This causes the standard deviation to momentarily increase



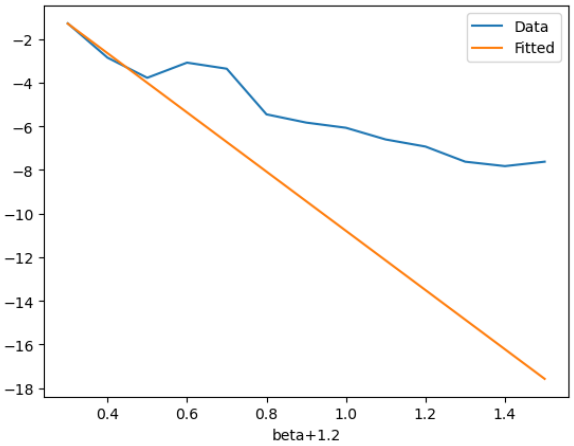
(a) Final coverage mean, Nonlinear Space



(b) Final coverage mean, Linear Space



(c) Final coverage std., Nonlinear Space



(d) Final coverage std., Linear Space

Figure 4.8: Nonlinear fittings of the two coverage criteria vs. process time. The orange line is the predicted coverage, and the blue line is the actual coverage.

before settling back down.

4.4.3 Estimated Weight Moving Average Method

Even with a highly accurate process model, inherent noise within the system or process disturbances that shift the process model may affect the R2R controller's ability to maintain the system at the desired setpoint. One widely used methodology to mitigate these challenges is the Exponentially Weighted Moving Average (EWMA) method, which updates the $\alpha_c,\ \beta_c$ tuning parameters in Eq. (4.6) by taking the exponentially weighted moving average of its past values. This effectively gives the controller information regarding its past error, which allows it to adjust and overcome the above challenges.

In real-world applications, the slope α_c is typically assumed to remain constant, even under various disturbances, while the intercept β_c is set to be adjustable [28]. Thus, the updating mechanism for the intercept β for process metric c is defined by the following equation [14]:

$$\beta_{c,i+1} = (1-\lambda)\beta_{c,i} + \lambda(\psi_{c,i} - \alpha_c \chi_{c,i}) \tag{4.8}$$

where $\beta_{c,i+1}$ is the updated intercept, $\beta_{c,i}$ is the intercept used in the previous run, $\psi_{c,i}$ is the transformed output of the previous run, α_c is the slope, and $\chi_{c,i}$ is the transformed input of the previous run. As all of the terms for the previous run are already known, $\beta_{c,i+a}$ can be easily solved for each controller as follows.

$$\beta_{m,i+1} = (1-\lambda)\beta_{m,i} + \lambda\left(\ln\frac{cov_{m,i}}{1-cov_{m,i}} - \alpha_m \ln\left(\delta_{m,i} + 0.75\right)\right)$$

$$\beta_{s,i+1} = (1-\lambda)\beta_{s,i} + \lambda\left(\ln cov_{s,i} - \alpha_s(\delta_{s,i} + 0.75)\right)$$

Then, $\beta_{c,i+1}$ can be plugged into Eqs. (4.6) and (4.7) to find $\chi_{c,i}$ and subsequently $\delta_{c,i+1}$.

The exact solution for the final coverage mean ($c = m$) is shown below.

$$\ln \frac{cov_{m,d}}{1 - cov_{m,d}} = \alpha_m \gamma + \beta_{m,i+1},\ \gamma = \ln\left(\delta_{m,i+1} + 0.75\right) \tag{4.10a}$$

$$\gamma = \left(\ln \frac{cov_{m,d}}{1 - cov_{m,d}} - \beta_{m,i+1} \right) / \alpha_m$$

$$\delta_{m,i+1} = e^{\gamma} - 0.75 \tag{4.10b}$$

where Eq. (4.10a) is the full form of Eq. (4.6) with the nonlinear m terms from Eq. (4.7) substituted in, Eq. (4.10b) is the final equation that the R2R controller uses to find the process time offset, $cov_{m,d}$ is the desired final coverage mean, γ is a placeholder variable as defined in Eq. (4.10a), and $\delta_{m,i+1}$ is the process time offset for the next run as determined by the final coverage mean R2R controller. Similarly, the exact solution for the final coverage std. ($c = s$) is as follows:

$$\ln cov_{s,d} = \alpha_s(\delta_{s,i+1} + 0.75) + \beta_{s,i+1} \tag{4.11a}$$

$$\delta_{s,i+1} = \frac{\ln cov_{s,d} - \beta_{s,i+1}}{\alpha_s} - 0.75 \tag{4.11b}$$

where Eq. (4.11a) is the full form of Eq. (4.6) with the nonlinear s terms from Eq. (4.7) substituted in, Eq. (4.11b) is the final equation that the R2R controller uses to find the process time offset, $cov_{s,d}$ is the desired final coverage std., and $\delta_{s,i+1}$ is the process time offset for the next run as determined by the final coverage mean R2R controller.

Once both controllers have found their respective δ_c, the final δ_f is set as the largest of the two to minimize underprocessing.

$$\delta_{f,i+1} = \max(\delta_{m,i+1}, \delta_{s,i+1})$$

where $\delta_{f,i+1}$ is the final process time offset that is used to determine the process times of the next run, $\delta_{m,i+1}$ is the process time offset found by the final coverage mean controller, and $\delta_{s,i+1}$ is the process time offset found by the final coverage std. controller. Then, Eq. (4.12) is used to find the

process times for the next run.

$$t_{A,i+1} = t_{A,0} + \delta_{f,i+1} \tag{4.12a}$$

$$t_{B,i+1} = t_{B,0} + \delta_{f,i+1} \tag{4.12b}$$

where $t_{A,i+1}$ is the process time of the next HF reaction, $t_{A,0}$ is the initial process time for the HF reaction, $t_{B,i+1}$ is the process time of the next TMA reaction, and $t_{B,0}$ is the initial process time for the TMA reaction.

4.5 Endpoint Controller Results and Analysis

4.5.1 Endpoint Controller Testing Dataset

The EP controller's main objective is to accurately stop the process. To evaluate its ability to do so, the EP control system is tested on complete runs with different parameters from the training simulations; these runs are called testing runs to differentiate them from the training/validation data used to develop the transformer model. The main difference between the testing data and the training/validation data is that the testing data is examined in real time rather than separated into multiple input sequences. Specifically, every 0.1 s, the EP controller receives the real-time wafer surface pressure data and makes a decision about whether to terminate or continue the process. The point at which the EP controller first decides to terminate the process is referred to as t_{ep}, and it represents the end time as determined by the EP controller. For the sake of analysis, each process run is simulated to 3.0 s regardless of the t_{ep}. This is so that the optimal end time, t_{tr}, can be determined and compared to t_{ep}.

An important note is that undershooting the predicted end time ($t_{ep} < t_{tr}$) is much worse than overshooting ($t_{ep} > t_{tr}$). When the model overshoots, it effectively makes the process run longer than what is necessary. While this wastes some reagent and time, the wafer is still successfully

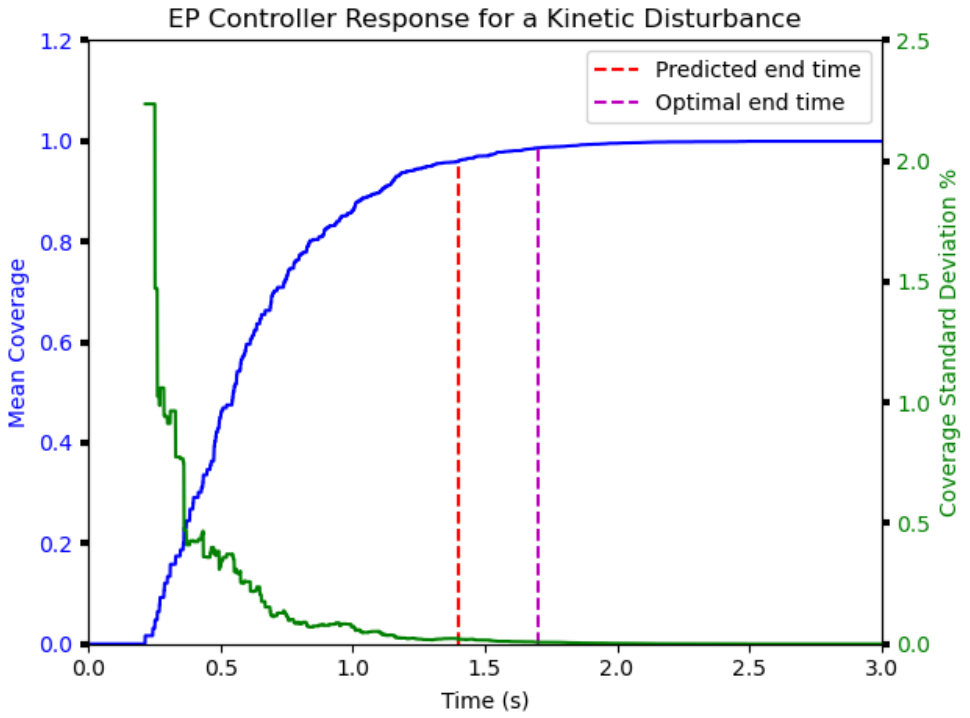


Figure 4.9: The blue line represents the final coverage mean throughout the run, the green line is the final coverage std., the vertical dotted red line is t_{ep}, and the vertical dotted purple line is t_{tr}.

processed. Conversely, when the model undershoots, the wafer is underprocessed and will most likely have to be thrown away. Thus, undershooting is much worse than overshooting. For this reason, the error metric is weighted so that undershooting is more heavily penalized.

$$e = \begin{cases} |t_{ep} - t_{tr}| & \text{if } t_{ep} > t_{tr} \\ 2|t_{ep} - t_{tr}| & \text{otherwise} \end{cases} \tag{4.13}$$

where e is the error metric used to evaluate the EP system's performance, t_{ep} is the end time predicted by said model, and t_{tr} is the optimal end time. For example, in Fig. 4.9, the predicted end time is 1.4 s while the true end time is 1.7 s. Thus, the error associated with this run is 0.6.

Section 4.5.1 will be used to evaluate the EP controller's efficacy at mitigating various process disturbances in two ways: first is its robustness, or how changing the training data themselves affects the model performance. Then, the controller is evaluated on its consistency, which is how noise in the training data affects the EP control system.

4.5.2 Robustness

To understand how the training data affects the model performance when under various disturbances, multiple EP controller are first trained on the datasets described in Section 4.3.2 that have datasets with either pure kinetic, pure pressure, or both kinetic and pressure disturbances. Note that the kinetic disturbance is directly applied to the reaction rate k rather than to any individual constant, which was the case for the training data. Then, each EP controller is run on testing datasets with the same set of disturbances, and the error as described in Section 4.5.1 is calculated for each run. The results for Reaction A are shown in Table 4.4 and the results for Reaction B are shown in Table 4.5.

Generally speaking, the EP controller improves when it is trained on the types of disturbances that it is tested on. For example, when the system is trained on a kinetic disturbance from Reaction

Table 4.4: Error comparison for Reaction A with a Kinetic/Pressure spread.

Training Data	**Validation Data**		
	Kinetic	Pressure	Both
Kinetic	0.467	1.060	0.915
Pressure	0.593	0.313	0.285
Both	0.613	0.413	0.440

Table 4.5: Error comparison for Reaction B with a Kinetic/Pressure spread.

Training Data	**Validation Data**		
	Kinetic	Pressure	Both
Kinetic	0.153	1.707	1.790
Pressure	0.733	0.164	0.250
Both	0.373	0.267	0.310

A, it performs better on the kinetic test ($e = 0.467$) compared to the pressure test ($e = 1.060$), and vice versa. Of note, while it is vital to train the model on the disturbances it is expected to face, the pressure disturbance has a much larger impact than the kinetic disturbance. For both Reaction A and Reaction B, when examining the column where the validation data has both kinetic and pressure disturbances, the model trained on only pressure disturbance data outperforms all others. This indicates that the pressure disturbance plays a much larger role in the model's ability to understand the process than the kinetic disturbance, to the point where a model trained on only the pressure disturbance outperforms a model trained on both, even when validated on a dataset with both disturbances. This shows that the optimal strategy for training an EP controller is to simply use training data similar to what the model will be used on. In industry, this is trivial as each process has many years of data [106].

It is worth mentioning that the impact of the pressure disturbances may not come from it affecting the actual kinetics of the reactions; usually, kinetic disturbances have a greater effect on the actual reaction rates and t_{tr}. Rather, this phenomenon is most likely due to how the model uses the surface pressure of the wafer as its input. Additionally, the range of the surface pressure, ± 5 Pa, is

relatively narrow for each run compared to the scale of pressure disturbance, ± 100 Pa. This means that a good model must necessarily adapt to the wide range of pressure disturbances. This explains the poor performance of the models trained on kinetic data that are tested on runs with pressure disturbances; the pressure disturbances shifts the wafer surface pressure far beyond the pressure range that the model is used to, which makes all the input sequence's values seem abnormal.

4.5.3 Consistency

It is also important to understand how well the EP controller can handle noise, which is referred to as its consistency. Each reaction was run ten times, each with a randomly selected ν and σ, which are the same variables used in Table 4.2, and no other process disturbances. The value of these two variables were selected from a Gaussian distribution centered around 1.0 with a standard deviation of 0.1 because this distribution follows the industrial standard of an average fail rate of 2%. The results of the twenty total runs are shown in Tables 4.6 and 4.7.

Table 4.6: Evaluation of Reaction A with no disturbances and moderate noise. Average error is 0.47.

No.	ν	σ	t_{tr}	t_{ep}	**Error**
Run 1	0.939	1.020	1.3	1.3	0.00
Run 2	1.120	0.969	1.0	1.5	0.50
Run 3	0.922	1.030	1.0	1.5	0.50
Run 4	0.930	0.999	1.1	2.1	1.00
Run 5	0.947	0.923	1.1	1.7	0.60
Run 6	1.010	0.910	1.0	1.5	0.50
Run 7	1.160	0.938	1.8	2.0	0.20
Run 8	1.150	0.819	1.1	1.5	0.40
Run 9	0.934	1.140	1.2	2.1	0.90
Run 10	0.888	0.882	1.2	1.3	0.10

Reaction A has a higher average error ($e = 0.46$) compared to Reaction B ($e = 0.08$), which suggests that the model for Reaction A is not as accurate as the model for Reaction B. This result

Table 4.7: Evaluation of Reaction B with no disturbances and moderate noise. Average error is 0.08.

No.	ν	σ	t_{tr}	t_{ep}	**Error**
Run 1	1.080	0.916	1.2	1.2	0.00
Run 2	0.979	0.856	1.4	1.3	0.20
Run 3	1.090	1.050	1.2	1.3	0.10
Run 4	1.070	1.070	1.2	1.3	0.10
Run 5	0.877	1.000	1.4	1.6	0.20
Run 6	0.954	1.230	1.3	1.3	0.00
Run 7	1.060	1.080	1.2	1.3	0.10
Run 8	0.964	0.913	1.3	1.3	0.00
Run 9	0.946	0.995	1.3	1.4	0.10
Run 10	1.170	1.090	1.2	1.2	0.00

is corroborated by Tables 4.4 and 4.5, as the average error of a model when tested on the same dataset it was trained on is 0.407 for Reaction A and 0.209 for Reaction B. Both average errors are comparable to the ones in Tables 4.6 and 4.7; thus, the EP controller for Reaction A performs worse than that of Reaction B. But regardless of the model's baseline prediction ability, the results for Reaction B show that the EP method is resistant to noise when the model is trained on datasets where those parameters are varied.

4.6 Run-to-Run Controller Results and Analysis

4.6.1 Run-to-Run Environment

The EP controller shows great potential for mitigating common disturbances in an industrial manufacturing environment. However, the controller may not be possible to implement for all processes, as the controller for Reaction A was not as consistent as that of Reaction B even though they were trained on similar datasets. Thus, there is still motivation to design more complex control schemes for processes that are hard for data-driven machine learning models to learn.

A common control system for ALE processes is ex-situ run-to-run (R2R) control, which adjusts process parameters after directly measuring the process outcome after the process is complete. Thus, this section examines how EP and R2R control systems can work together under various frameworks. Specifically, all the control systems will be tested under the same conditions; the process will experience a sudden shift where all the kinetic activity is lowered by 40%. This is a relatively large in comparison to the shifts examined for the pure EP controller, and any failure to account for this shift will result in scrapped material and wasted time. The ideal combination of control systems will quickly adjust the process time so that the final coverage criteria are met with minimal over-processing.

The control systems will be evaluated on how many runs they take to return to the target final coverage criteria and how much overprocessing occurs. Because there are two final coverage criteria, a total of three error calculations are made:

$$\epsilon_m = \sum_{i=1}^{L} c_{m,i} \cdot \frac{|cov_{m,i} - 0.96|}{L}, \quad \text{where } c_{m,i} = \begin{cases} 1 & \text{if } cov_{m,i} \geq 0.96 \\ 2 & \text{if } cov_{m,i} < 0.96 \end{cases} \tag{4.14a}$$

$$\epsilon_s = \sum_{i=1}^{L} c_{s,i} \cdot \frac{|cov_{s,i} - 0.02|}{L}, \quad \text{where } c_{s,i} = \begin{cases} 1 & \text{if } cov_{s,i} \leq 0.02 \\ 2 & \text{if } cov_{s,i} > 0.02 \end{cases} \tag{4.14b}$$

$$\epsilon_t = \sum_{i=1}^{L} 0.01 \cdot \frac{t_{A,i} + t_{B,i}}{L} \tag{4.14c}$$

where ϵ_m is the error term associated with the final coverage mean criterion, $c_{m,i}$ is a scaling factor based on if the final coverage mean criterion was met for run i, $cov_{m,i}$ is the final coverage mean of run i, L is the number of process runs, ϵ_s is the error term associated with the final coverage std. criterion, $c_{s,i}$ is a scaling factor based on if the final coverage std. criterion was met for run i, $cov_{s,i}$ is the final coverage std. of run i, ϵ_t is the error term associated with overprocessing, $t_{A,i}$ is the process time for the HF reaction for run i, and $t_{B,i}$ is the process time for the TMA reaction for

run i. The final, comprehensive error term is found by simply summing up the three error terms of Eq. (4.14) as shown below:

$$\epsilon_f = \epsilon_m + \epsilon_s + \epsilon_t \tag{4.15}$$

where ϵ_f is a comprehensive error term used to evaluate the various control systems presented in this chapter.

Of these control systems, first is a standalone EWMA-R2R system that is meant to establish a baseline expectation for the following control systems. Second, the EP system will be evaluated on its own to better compare real-time and ex-situ controllers. Finally, combined systems will be examined: an EP+SCC system, and an EP+EWMA system. “SCC" stands for Standard Case Corrector, which is a newly developed R2R ex-situ controller. All of these systems will be compared through two metrics. First and foremost, they will be evaluated on how many wafers are scrapped, or thrown away; this occurs when the final coverage criteria is insufficient. If two control systems have the same number of scrapped wafers, then they will be evaluated on how much time they waste on overprocessing. With these metrics, the best control system among the aforementioned four systems can be determined.

4.6.2 Pure EWMA Controller

The EWMA-R2R controller processes coverage data by first converting the measured mean and standard deviation to nonlinear forms, then applying the EWMA algorithm to determine the process time. While Eqs. (4.10) and (4.11) describe how the EWMA-R2R controller updates the run parameters after each run, they do not describe the initial starting point of the process system. In this chapter, the initial process times, $t_{A,0} = 0.75s$ and $t_{B,0} = 1.05s$, are set to achieve a final coverage whose mean is over 96% and std. is less than 2% when there are no disturbances. Although the ALE process has two half-reactions, only the final etch per cycle (EPC) is measurable, preventing the R2R controller from adjusting each half-reaction individually. Instead, both t_A and

t_B are updated simultaneously using a process time offset δ, adjusted by the controller as shown in Eq. (4.12) and restated here.

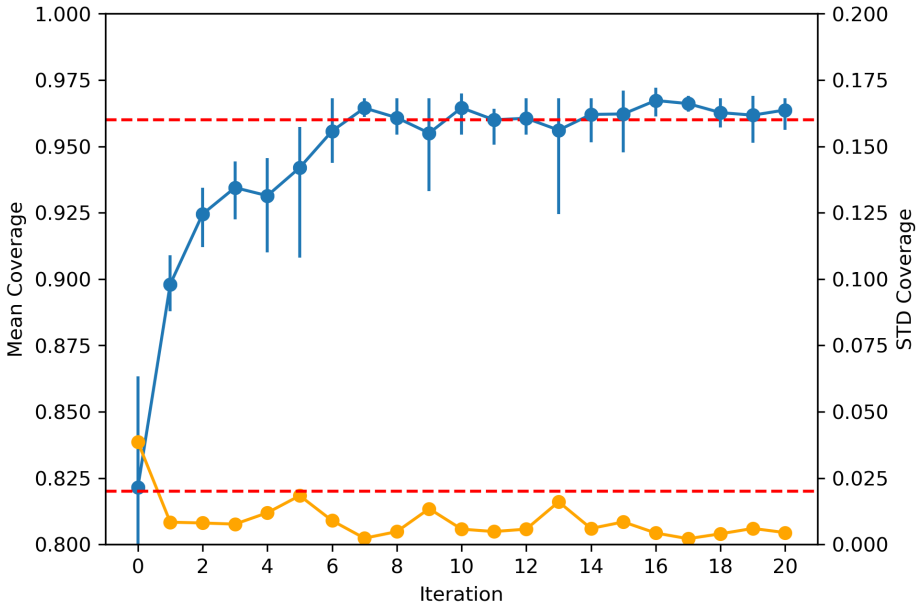
$$t_A = t_{A,0} + \delta$$

$$t_B = t_{B,0} + \delta$$

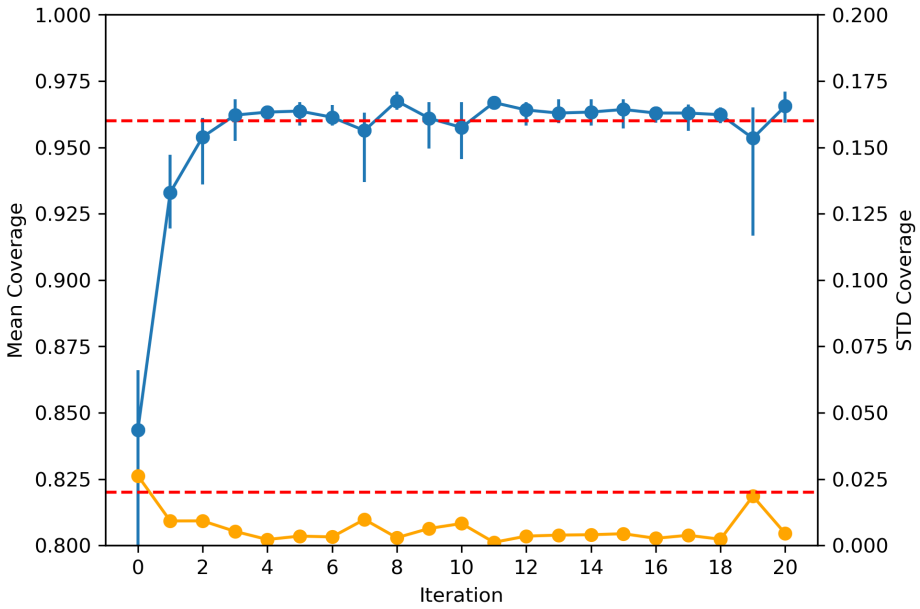
where t_A and t_B are the process times for the HF and TMA reactions, respectively, $t_{A,0}$ and $t_{B,0}$ are their initial process times, and δ is the time offset determined by the R2R controller.

The performance of the EWMA-R2R controller depends on two factors: the accuracy of the process model used in Eq. (4.6), and the value of λ in Eq. (4.8), which is a tunable factor that determines how much weight is given to recent measurements. A larger λ value makes the controller more responsive to recent batches, enabling quicker, more aggressive corrections, while a smaller λ emphasizes historical data, leading to a more conservative, stable response. Although aggressive settings can correct shifts faster, conservative settings reduce the risk of oscillations or divergence. This study uses λ values of 0.7 for aggressive and 0.3 for conservative control.

Performance results for two pure EWMA-R2R controllers with $\lambda = 0.3, 0.7$ are shown in Figs. 4.10a and 4.10b. These plots show that the EWMA-R2R controller can drive the system back to the desired setpoint even with a large process shift, where all kinetic rates are reduced by 40%. The more aggressive $\lambda = 0.7$ controller in Fig. 4.10b results in fewer scrapped wafers compared to the conservative $\lambda = 0.3$ controller in Fig. 4.10a because the aggressive EWMA controller reaches the final coverage criteria faster, indicating that it is better suited for this ALE process. This is also supported by the ϵ_f criterion, as the more aggressive EWMA has a lower ϵ_f of 0.061 compared to the conservative controller's 0.074. However, even though the aggressive EWMA controller performs well, it still has a key limitation in its inefficiency at the initial stages; several wafers are misprocessed before the process fully corrects.



(a) Pure EWMA, $\lambda = 0.3$; 8 scrapped wafers, $\epsilon_f = 0.074$



(b) Pure EWMA, $\lambda = 0.7$; 5 scrapped wafers, $\epsilon_f = 0.061$

Figure 4.10: Control results of the EWMA-R2R controller. The blue line is the mean coverage, the orange line is the std. coverage, the high red dashed line is the mean coverage target, and the low red dashed line is the std. coverage target.

4.6.3 Pure Endpoint Controller

The EP controller is a real-time controller, which means that there is no parameter updating or changing in the controller in between each run. Thus, the process times are represented by the following equations:

$$t_A = EP_A$$

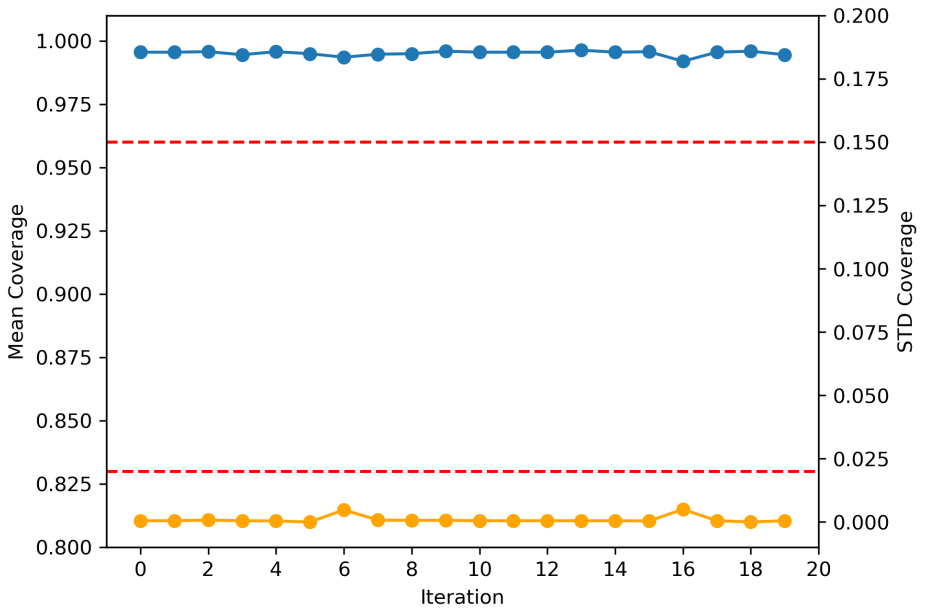
$$t_B = EP_B$$

where t_A and t_B are the process times for the HF and TMA reactions, respectively, and EP_A and EP_B are the process times as determined by the EP controller for the HF and TMA reactions, respectively.

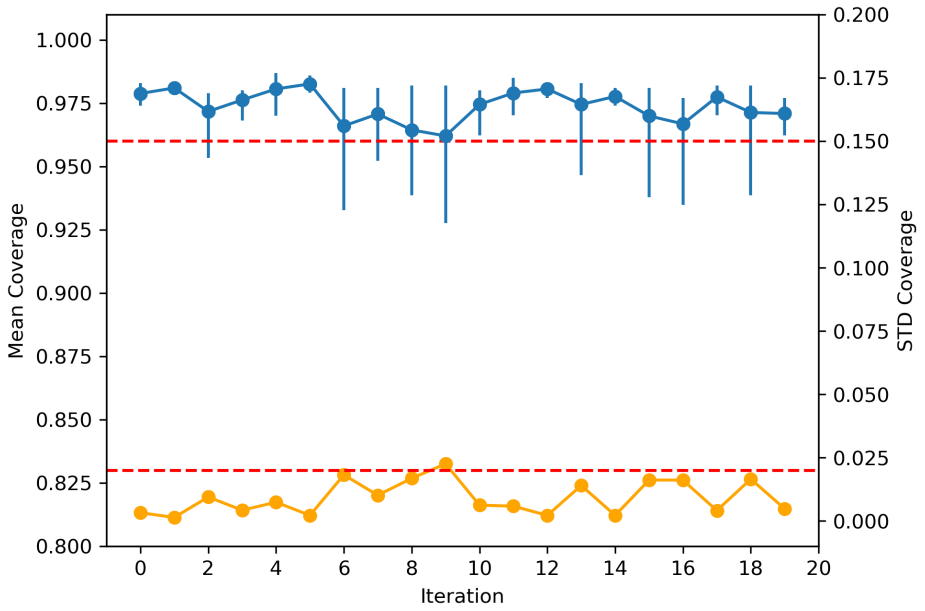
Rather, all the differences in between the runs stem from the stochastic nature of the multiscale simulations and the Transformer model in the EP controller. The EP controller used for this simulation is the same one described in Section 4.3, but the threshold of the final sigmoid function is adjusted; this represents the transformer's confidence that the process has terminated. Two thresholds were considered: a conservative EP controller with a sigmoid threshold of 0.96 for Reaction A and 0.99 for Reaction B, and an aggressive EP controller with a threshold of 0.5 for both reactions. Fig. 4.11a shows the performance of the conservative EP controller and Fig. 4.11b that of the aggressive controller.disturbances.

The conservative and aggressive pure EP controllers in Figs. 4.11a and 4.11b both resulted in 0 scrapped wafers. However, the conservative EP controller consistently resulted in large amounts of overprocessing, causing its ϵ_f of 0.099 to be above even that of the pure EWMA controllers. In comparison, the aggressive EP controller has the lowest ϵ_f of all four controller systems at 0.056. This means that it is best suited for handling sudden, large process shifts, and its performance here highlights its ability to reduce precursor usage and improve manufacturing efficiency.

Both EP controllers have some overprocessing, but they are still able to prevent all mispro-



(a) Conservative EP Controller, Threshold = 0.95 (HF)/0.99 (TMA); 0 scrapped wafers, $\epsilon_f = 0.099$



(b) Aggressive EP Controller, Threshold = 0.5 (HF)/0.5 (TMA); 0 scrapped wafers, $\epsilon_f = 0.056$

Figure 4.11: Control results for pure EP controllers. The lines are the same as in Figs. 4.10a and 4.10b

cesses, even when the disturbance first appeared. In comparison, the pure EWMA-R2R controller requires several batches to adjust to the disturbance before there are no more misprocesses. Despite its advantages, the pure EP controller also has other weaknesses; it relies on time-series pressure profiles, which are influenced by the stochastic nature of surface reactions and noise in the measuring equipment. This causes it to predict different endpoint times even when the process conditions are identical. Thus, combining EP and R2R controllers can make up for their individual shortcomings. For the combined control systems shown next, the aggressive EP controller is used as it performed better than the conservative EP controller.

4.6.4 Standard Case Corrector

As both the EP and R2R controllers use the process time as their control variable, there are many ways to combine the two systems. As mentioned earlier, while the EP controller does prevent misprocessing, it can be volatile when used for multiple runs. Thus, one combined EP+R2R method is to use a Standard Case Corrector (SCC) Controller. This controller assumes that the process time required to adjust one set of coverage criteria to another is the same regardless of if there are any disturbances.

As illustrated in Fig. 4.12, after each run, the SCC controller uses the final coverage mean and std. progression curves of a standard case without any disturbances to find two key values for each curve: t_0, which is the time needed to reach the target set point, and t_m, which is the time needed to reach the measured output of the most recent run. The controller then calculates the sum $t_d = t_m - t_0$. Like $\beta_{f,i+1}$ in Eq. (4.12), the final t_d is the maximum between the one calculated from the final coverage mean curve and the one calculated from the final coverage std. curve. Additionally, the fitted progression curves of Section 4.4.2 are used as the standard case examples. This is because the smoother curves prevent the controller from forming any offsets. If the true final coverage criteria progression curves were used, their non-monotonic nature could cause issues in the SCC controller. When a disturbance is sensed, the system is "reset" with a run

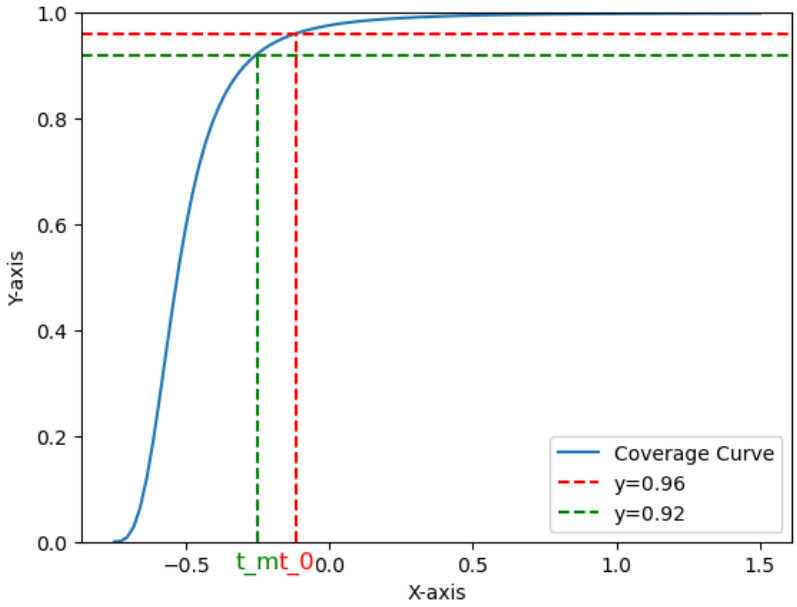


Figure 4.12: Representation of how the SCC Controller calculated t_d for the final coverage mean.

that only uses the endpoint controller. This effectively sets $t_{A,0} = EP_A$ and $t_{B,0} = EP_B$. Then, the process time continues to be updated by modifying Eq. (4.12) as follows:

$$t_d = t_m - t_0 \tag{4.18a}$$

$$t_{A,i+1} = t_{A,i} + t_d \tag{4.18b}$$

$$t_{B,i+1} = t_{B,i} + t_d \tag{4.18c}$$

where t_d is the value found in Fig. 4.12, $t_{A,i}$ and $t_{B,i}$ are the most recent process times for reactions A and B, and $t_{A,i+1}$ and $t_{B,i+1}$ are the next set of process times for reactions A and B. Note that, unlike the EWMA-R2R controller, the SCC-R2R controller does not rely on a linear model. Thus, it avoids using any nonlinear transformations, making it easier to implement.

The result of combining the EP and SCC controllers is shown in Fig. 4.13. Run 0 is where the disturbance is first introduced, resulting in a misprocess. Run 1 is the pure EP run, and Runs 2 and onwards are the SCC-controlled runs as defined by Eq. (4.18). Even though the EP+SCC control system results in a scrapped wafer in Run 0, it successfully brings the process back within control

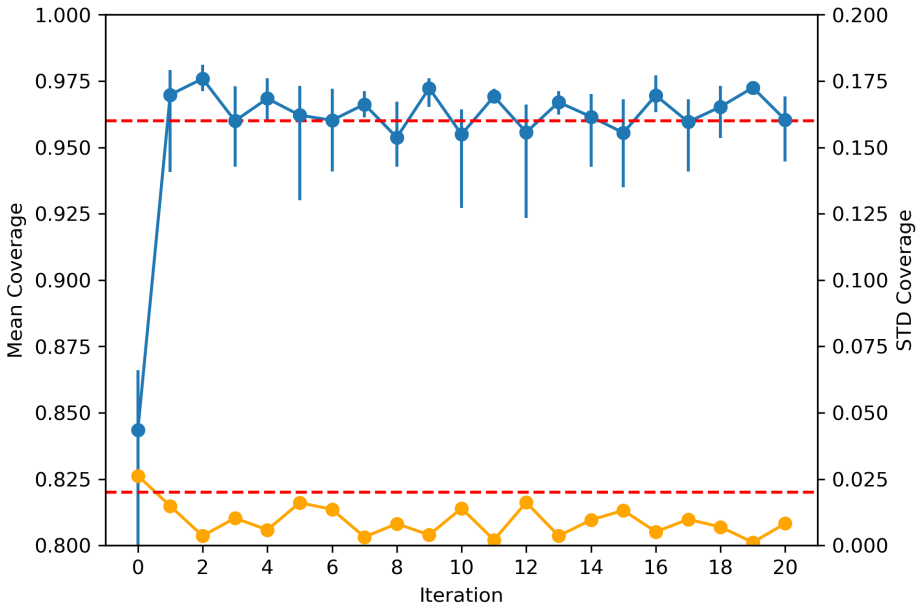


Figure 4.13: Control results for the EP+SCC controller; 6 scrapped wafers, $\epsilon_f = 0.057$.

by Run 1 and continues to tightly control the final coverage criteria around the target set point due to the unstable nature of the process. Thus, the EP+SCC is insufficient to control the process.

While the EP+SCC controller may be worse than the pure EP controller when it comes to sudden, unexpected process disturbances, that is not always the case. In manufacturing environments, it is common practice to have qualifying test runs after a major equipment cleaning in order to detect any process shifts. In this scenario, Runs 0 and 1 of the EP+SCC controller can be thought of as qualifying runs that do not count towards the misprocessing rate. In that case, the ϵ_f should only span Runs 2-20. When evaluated in this context, the EP+SCC controller's ϵ_f becomes 0.047, which is lower than the aggressive EP controller's ϵ_f of 0.056. Thus, in the right circumstances, the EP+SCC controller can outperform the pure EP controller.

4.6.5 EWMA and EP controller

While the EP+SCC controller has a better performance than either of the controllers on their own in a controlled manufacturing environment, many of the batch runs did not meet the final cov-

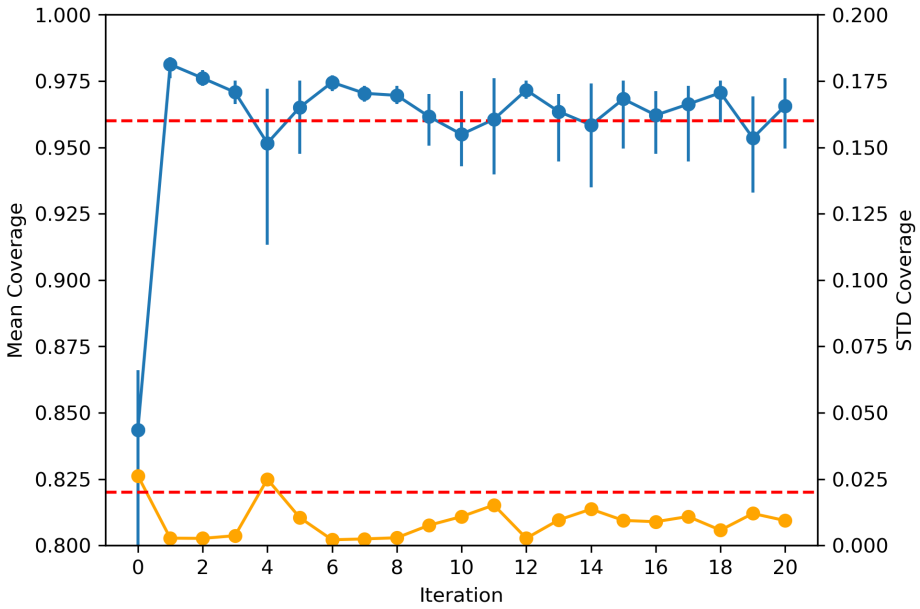


Figure 4.14: Control results for the EP+EWMA controller; 4 scrapped wafers, $\epsilon_f = 0.060$.

erage criteria. Thus, the combined controller can still be improved upon, and other combinations of R2R and EP controllers must be explored. In this section, the EWMA-R2R controller in Section 4.6.2 is combined with the EP controller in Section 4.6.3 to create another hybrid approach.

While both the EWMA-R2R and EP controllers use the process time as their control variable, there is no issue as the combined controller functions similarly to the SCC controller; once a disturbance is detected, the process times are reset after a run that only has an EP controller. Once that run is completed, the control system reverts to the classical EWMA-R2R equations of Eqs. (4.10) to (4.12). The results of combining a EWMA-R2R controller with $\lambda = 0.7$ and an aggressive EP controller are shown in Fig. 4.14. Run 0 is where the disturbance is first introduced, resulting in a misprocess. Run 1 is the pure EP run, and Runs 2 and onwards are controlled by the same EWMA-R2R controller discussed earlier in Section 4.6.2.

Like the EP+SCC controller, the EP+EWMA controller system performs worse than the pure EP system when it comes to reacting to a sudden process shift. However, when evaluated in a well-controlled manufacturing environment, its ϵ_f becomes = 0.049. While this is still larger than

the EP+SCC controller's ϵ_f of 0.047, which indicates that the EP+SCC controller is ideal, it is nonetheless higher than the pure EP controller's ϵ_f of 0.056.

The final results of all the different control systems are summarized in Table 4.8. For manufacturing environments that are poorly controlled with a potential for unexpected process shifts, the pure EP control system is the best option as it has the lowest ϵ_f when all the runs are considered. However, for a manufacturing environment that is well controlled and where process shifts only occur at known events, the combined systems are better. Of the two, the EP+SCC controller has a slightly lower ϵ_f when only Runs 2-20 are considered. Thus, the pure EP controller is best suited for poorly controlled manufacturing environments, and the EP+SCC controller is best suited for well controlled environments.

Table 4.8: Summary of the R2R evaluation criteria.

	ϵ_f, Runs 0-20	ϵ_f, Runs 2-20
EWMA, $\lambda = 0.3$	0.074	-
EWMA, $\lambda = 0.7$	0.061	-
EP, aggressive	0.056	-
EP, conservative	0.099	-
EP+SCC	0.057	0.047
EP+EWMA	0.060	0.049

4.7 Conclusions

This chapter presents an integrated control strategy combining a real-time endpoint (EP) feedback controller, based on a transformer machine learning architecture, with an ex-situ Run-to-Run (R2R) controller for an Al_2O_3 atomic layer etching (ALE) process. The EP controller was trained with simulated process data and then tested on a different set of simulated process data for two different metrics: robustness and consistency. This novel controller enables real-time detection of process indicators for the ALE process, effectively handling kinetic and pressure disturbances. For R2R control, this chapter introduced a new EWMMA strategy involving nonlinear transformations

to create a linear relationship and a novel standard case corrector (SCC) simplified the overall implementation by eliminating the need for complex nonlinear modeling.

Various combinations of EP and R2R controllers were applied to the ALE process under a severe negative kinetic disturbance. Two manufacturing environments were considered: a poorly controlled environment where process shifts occur randomly and without warning, and one where process shifts are expected (e.g., after maintenance is done on the etching tool). For the former case, the pure EP controller performed best, as it had the lowest error metric, $\epsilon_f = 0.056$, when considering all runs, including the initial disturbance run. But for the latter case, only Runs 2-20 are used to calculate ϵ_f as Runs 0 and 1 are considered to be qualifying test runs used to adjust the process parameters. In that case, the EP+SCC controller performed the best as it had the best performance of $\epsilon_f = 0.047$ at maintaining the system at the desired setpoint after the kinetic disturbance was implemented. This hybrid approach leverages the strengths of both controllers, offering a significantly improved performance over the traditional pure EWMA and pure EP controllers.

The controllers developed in this chapter only use the surface wafer pressure in their machine-learning models, but in reality, the amount and variety of process data that is available in a high-volume manufacturing environment is many times larger. Whether it be incorporating multiple data streams, aggregating these large datasets, or using bleeding edge machine-learning models, industrial manufacturing represents a space with abundant opportunities for innovation.

Chapter 5

Industrial Data-Driven Machine Learning Soft Sensing for Optimal Operation of Etching Tools

5.1 Introduction

In the current decade, there has been a world-wide surge in demand for electronic products. This has in turn dramatically increased the manufacturing demand for related commodities such as microelectronics, hard drives, and integrated circuits [58]. The innovations of modern-day electronics and their rising demand is in part owed to the rising density of transistors in semiconductor chips, which improves the computing performance of these chips [5]. This increased demand has resulted in recurring shortages of electronics and has hurt the global economy, which now depends on the manufacturing of electronic devices. Thus, there is a growing need to pursue innovation in the manufacturing sectors [57]. Smart manufacturing and Industry 4.0 concepts were proposed in the mid-2000s for developing smart plants and factories that utilize network communications, Information Technology (IT), Internet of Things (IoT) and big data [10, 21]. Industry 4.0 aims to

achieve resilient manufacturing processes that are characterized by high efficiency, high conformance, and high fidelity.

Smart Manufacturing, or Industry 4.0, necessitates holistic and continuous sensing, monitoring, and automation of processes that produce data in the form of quantifiable parameters. For instance, numerous advanced sensors are employed in manufacturing processes to monitor and understand the process parameters, and at the same time, they provide measured feedback to controllers for process automation. These sensors directly measure physical properties to obtain process information for monitoring and product quality assessment. For example, a quartz crystal microbalance is commonly used in the semiconductor industry to measure film thickness, providing coverage data for etching and deposition processes [77]. However, the operation of the measurement equipment is labor-intensive, time-intensive, and costly. In some scenarios, the capital and resource expenditures outweigh the value of the final product, one example of which is the treatment of wastewater [90]. However, the costs of these measurement steps can be mitigated through advanced process monitoring [65]. As industrial processes become increasingly complex, the direct measurement of key process parameters, which are often key performance indicators (KPIs), becomes more challenging. One process monitoring method is soft sensing, which detects critical process parameters by leveraging the wide range and scope of operational data that is generated in modern manufacturing processes [62]. Soft sensors use data from existing physical sensors and prior knowledge to develop data-driven algorithms that predict specific physical quantities and product quality, offering a more efficient, high-measuring frequency, and less labor-intensive alternative to traditional sensing approaches. Thus, they are particularly effective when there is a need to capture complex physical phenomena that are not easily measured or modeled or when there is a need for embedded fidelity that comes from years of operational experience. However, even when vast amounts of data are generated, the sensor performance will suffer if that data does not adequately cover the range, scope, and function of the operation relative to the sensor objectives.

In the context of modern complex manufacturing processes and the vast amounts of data they

generate, there is a growing body of research that applies machine learning methods to develop soft sensors for detecting process and product properties. Recent literature reviews on deep learning methods in soft sensing [81] emphasize the significance of neural network approaches, including Convolutional Neural Networks (CNN) [11], Recurrent Neural Networks (RNN) [43], and a combination of RNN and Feedforward Neural Networks [27]. Seagate Technology also reported using the novel transformer network to predict the PASS/FAIL of an industrial etching process with time series data as the input [106]. Deep learning with neural networks have advantages in capturing complex nonlinear correlations between input and output parameters, and when large amounts of training data are available, they often outperform traditional machine learning methods [72].

While many works have investigated which models are best suited for which deep-learning tasks, the question of if and how data aggregation can be used to supplement modeling tasks with small data volumes remains unanswered. Thus, this chapter proposes a deep-learning-based soft sensor developed using industrial data for detecting both binary properties (PASS/FAIL) and numerical properties (oxide thickness) for several industrial etching tools from Seagate Technology with high-dimensional input parameters. To address the problem where there is not enough data to train a model with high performance on a single tool, this chapter proposes a novel data aggregation method where datasets from other tools are combined with the dataset of a single tool to improve model performance on that specific tool. The data aggregation approach aims to improve the performance of the trained soft sensor model by properly (in a sense to be made clear below) combining datasets to increase the amount and variety of training data. Specifically, this chapter introduces a statistical method to optimize dataset selection during the aggregation process to improve aggregation efficiency.

This chapter is organized as follows: Section 5.2.1 provides an overview of the industry data used, Section 5.2.2 describes the preprocessing operations applied to the datasets, Section 5.2.3 and Section 5.2.4 describe the development of the soft sensor models, Section 5.3 demonstrates and evaluates the performance of the trained soft sensor models, and Section 5.4 summarizes the

findings of this chapter.

5.2 Data Processing and Modeling

This section describes the collection, processing, and contextualization of data from five industrial plasma etching tools, which are used to train two models: a classification model and a regression model. Then, we cover a cross-process data aggregation procedure for improving the classification model and the various loss functions used in training the regression model.

In the semiconductor fabrication industry, all products begin as a raw silicon wafer substrate. These substrates follow a set of procedures called the process flow, which describes each process step that the wafer must undergo. Once the wafer has gone through the entire process flow, it is a completed product. As the fabrication process is very repetitive, each wafer will be processed on the same tool multiple times at different process steps. This chapter examines a toolset of five electrically-induced plasma etching tools. This toolset consists of five physically identical chemical etching reactors that possess up to two chambers; the reactor is referred to as the "tool," and the chamber is referred to as a "module." Each module can run a variety of process steps. The process data is gathered on a per-module basis, which means that each datum is for a specific tool-module combination. For example, an entry from T1-PM1 means that the wafer was processed in module PM1 of tool T1. A detailed diagram explaining the overall manufacturing process is shown in Fig. 5.1.

5.2.1 Industrial Data Generation

The process data used in this chapter was collected from four tools: T2, T4, T5, and T7. Each tool has up to two modules: PM1 and PM2. Specifically, process data was collected from T7-PM1, T7-PM2, T2-PM2, T5-PM2, and T4-PM1. Each data entry comes from a single run, which is defined as a process step that starts at $t = 0$ and ends at a preset process time, t_{end}. During the

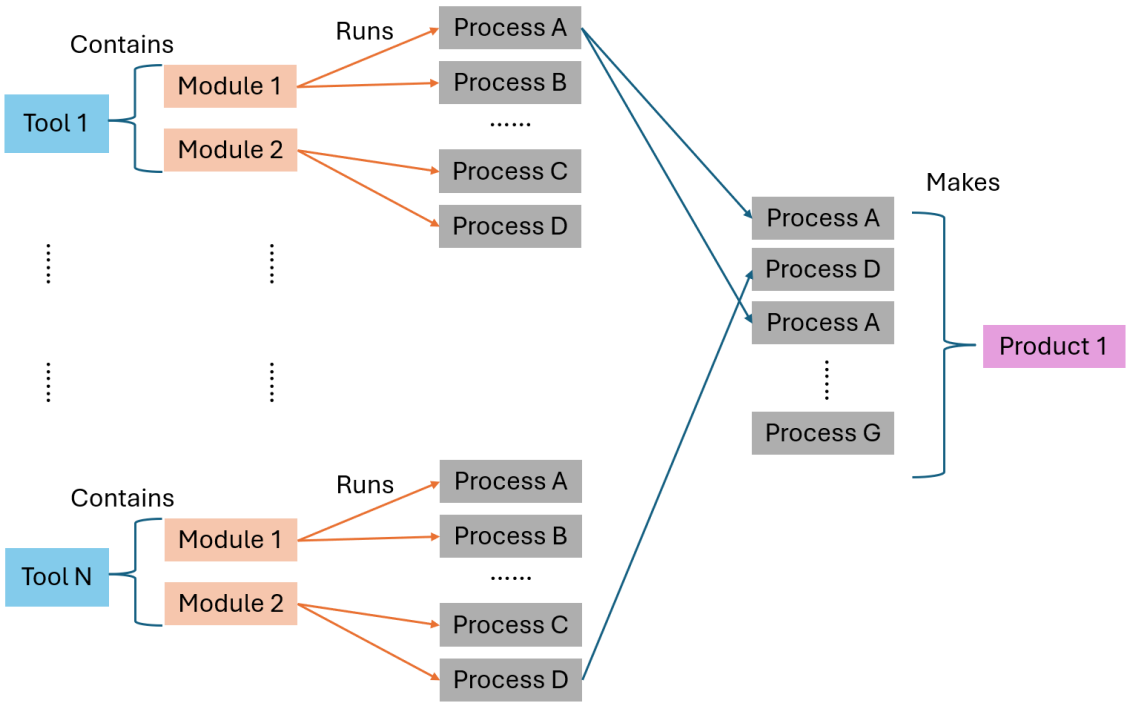


Figure 5.1: The overall manufacturing system for an industrial etching equipment. Each tool is an etching reactor that has multiple modules and can run various processes. Wafers start as pure silicon substrates, and after a series of production processes, they become a finished product.

process, physical sensors track 33 numerical features, and their average is recorded down alongside two process-specific discrete features and a time stamp. Thus, there is no time-series data. Once the wafer has finished processing, it is referred to as the product. The dataset used in this chapter spans from February 1st, 2018 to December 31st, 2022. The numerical features are categorized into three classes of variables which are pressure and gas flow, electromagnetic, and other equipment statuses and properties. The discrete features are the process name ID and substrate family ID, which are both alphanumeric text entries. These 35 pieces of information are inputs for both the classification and regression model.

The information that both models aim to predict is whether the process was successfully completed, and this information is gathered at a different tool. After the etch step, the product wafer is processed at a metrology tool that measures how much substrate was etched away. As all of the processes examined in this chapter are oxide etches, the metrology tool will measure the remaining oxide thickness. Depending on how much the measured oxide thickness deviates from the target oxide thickness, the run will be labeled as either a "PASS" or a "FAIL." The classification model uses the binary PASS/FAIL measurement as its output, and the regression model uses the target oxide thickness as an additional input and the measured oxide thickness as its output.

5.2.2 Data Preprocessing

Data preprocessing is a crucial step to effectively train any model. A properly preprocessed dataset allows the model to effectively learn the patterns of the data. For instance, [68] highlights the necessity and importance of data preprocessing in several deep learning-based applications. Preprocessing steps often include removing or filling in invalid and abnormal data as is appropriate, encoding discrete variables, and normalizing features to avoid skewing caused by the absolute value of variables. These steps ensure that the model receives consistent and relevant data, which facilitate better learning and prediction ability; specifically for interpolating unseen data points within the applied training range. The input data preprocessing procedures in this chapter are

demonstrated in Fig. 5.2, and the procedures for output data preprocessing are shown in Fig. 5.3.

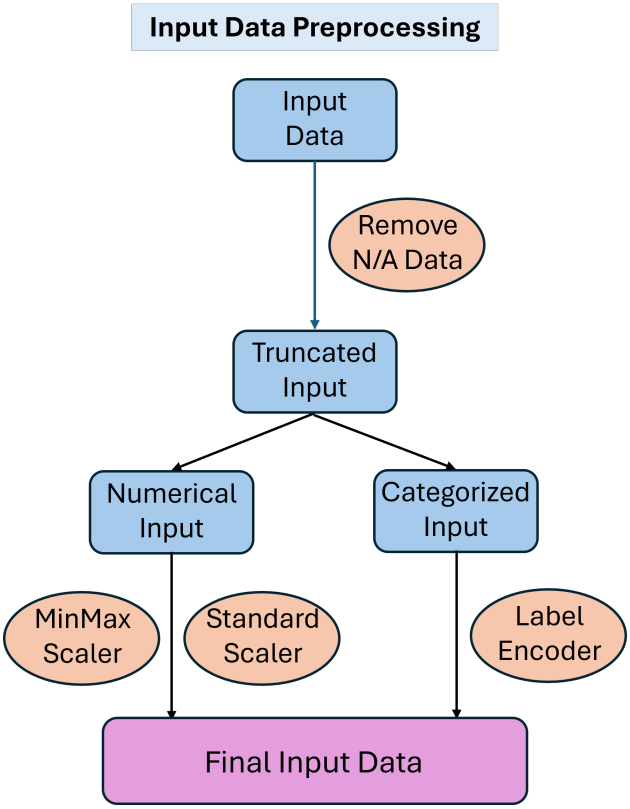


Figure 5.2: Input Data Preprocessing **Step 1**: Eliminate or pad missing data. **Step 2**: Scale the numerical features, and encode the discrete features. **Step 3**: Combine numerical and discrete features into one complete input dataset.

From a practical manufacturing standpoint, physical sensors do not function properly at all times, and not all parameters can be measured in all processes. Thus, there are almost always missing physical measurements in real industrial data. The data analyzed in this chapter is not exempt from this phenomena. Some particular tool-module combinations are missing entire features, and other features are only collected within certain time ranges. To address these issues, any feature that has no measured value (which is recorded as N/A) is filled in with a numerical value of 0 to maintain consistency with the other data points. On the other hand, any runs (one row of data) that are missing either of the outputs, which are the PASS/FAIL criterion and the measured oxide thickness, are removed from the dataset. Without the true results, the input features are meaning-

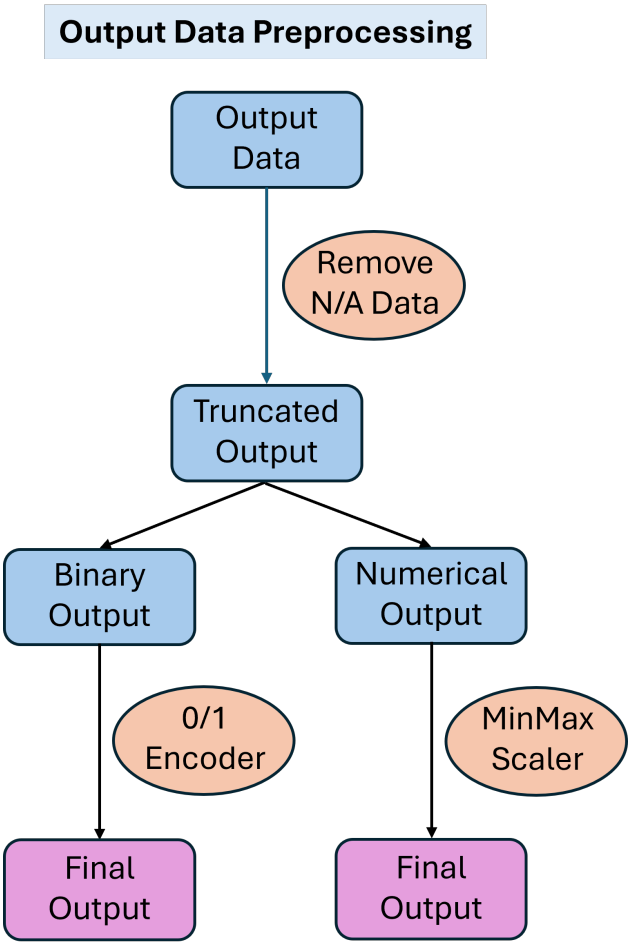


Figure 5.3: Output Data Preprocessing **Step 1**: Eliminate missing data. **Step 2**: Encode binary output to FAIL(0) and PASS(1). **Step 2-2**: Scale numerical output with MinMax Scaler.

less. By applying these two methods, all the invalid and abnormal data is pared from the datasets.

As with most machine learning models, which includes neural networks, all inputs must be numerical. Thus, it is necessary to encode any categorical or nonnumerical features if they are included in the training process. For this chapter, the label encoder from the scikit-learn package [64] is used to encode both the substrate family ID and the process name ID from alphanumeric features into numerical features. To create a consistent and holistic encoder that is capable of handling all possible cases, the encoder is trained on data that comprise the concatenation of all datasets from all tool-modules. This step creates a complete map for the encoder, ensuring that each discrete feature is transformed into a unique number, which avoids conflicts during the training of the model.

Additionally, for the binary PASS/FAIL output data, a 0/1 encoder is applied, which encodes PASS as 1 and FAIL as 0. These methods ensure that all the categorical features are transformed into numerical values so that the models can function effectively.

For neural networks, it is especially crucial to scale all numerical data to prevent the vanishing gradient and gradient explosion phenomena from occurring during the training process [69]. This is particularly important for input data features that vary significantly in absolute values. For instance, the dataset examined in this chapter has features in the range of both 10^{-4} and 10^{2}. A scaler normalizes the input features of the data, forcing each numerical feature to exist in a similar range and making the training process more stable and the optimizer task easier. For the classification task with a binary PASS/FAIL output, the numerical input features are scaled with a standard scaler, and the output features are already encoded as 1/0. For the regression task with numerical output features, separate MinMax scalers are applied for both the input and output data. The standard scaler is described in Eq. (5.1), and the MinMax scaler is described in Eq. (5.2). After the encoding process described earlier and the scaling processes described here are completed, all the input features, including the scaled numerical features and encoded categorical features, are concatenated into a data vector that represents a single run.

$$Z = \frac{X - u}{s} \tag{5.1}$$

$$Z = \frac{X - Min(X)}{Max(X) - Min(X)} \tag{5.2}$$

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. Note that the vectors used here consist of all the data for a particular feature (a data column). The standard scaler scales the original dataset around 0 for each feature in a similar range, while MinMax scaler scales the numerical data into a range of [0,1]. There is no fixed rule to determine the best scaling

method. Rather, scaler selection is dependent on the optimal model performance, which will be more rigorously defined later on in Section 5.3. Specifically, both the standard scaler and the MinMax scaler are applied to the same dataset and used to train two independent models. Then, the scaler that yields the best performance is chosen for that task.

5.2.3 Classification Model

The goal of a classification model is to accurately determine whether future, unknown wafers from a specific tool-module combination will pass or fail the following metrology step. The pre-processed input data vectors are the input variables to the model, and the output variable is a binary PASS/FAIL value. The dataset is first separated into two datasets by time: the modeling set and the test set. The modeling set comprises all the runs from February 1st, 2018 to December 31st, 2021, and it will be used to train the model. The test set comprises all the runs from January 1st, 2022 to December 31st, 2022, and it is used to evaluate model performance. These datasets are separated by time because it is necessary for the model to be generalizable across all times. From a practical point of view, the soft sensor model can only be considered successful if it can be effectively applied on unseen data and conditions from future manufacturing processes. To prevent overfitting, the modeling set is further separated into two more sets: the training set and the validation set. 80% of the modeling dataset is randomly chosen for the training set, and the remaining 20% is allocated to the validation set. The model is only trained and optimized on the training set, and the performance of each model is examined on both the training and validation sets to tune and optimize the generalization ability of the model. Stratified sampling is also used to ensure that the PASS/FAIL distribution is nearly identical between the training and validation datasets. With these specifications, when the model is trained on the training set and tested on the validation set, the candidate model is the model with the best performance on the validation set.

Model Training

The Feedforward Neural Network (FNN) is applied to the classification model in this chapter. The general structure of the network is shown in Fig. 5.4.

In an FNN, each neuron in the hidden layers takes a weighted sum of inputs from the input layer or previous hidden layers, applies a non-linear activation function, and then passes the result to the next layer. The final output layer produces the binary PASS/FAIL classification by using the sigmoid function to transfer the input value into the [0,1] range. The sigmoid function is also employed as the activation function in the hidden layers of classification models, due to its superior performance in classifier models [106]. The sigmoid function is described below in Eq. (5.3):

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{5.3}$$

The FNN has several tunable hyperparameters that need to be optimized to determine the best network structure. These hyperparameters include the number of hidden layers, the number of neurons in each hidden layer, the learning rate, and the L2 regularization coefficient. A grid search method is applied in this chapter to find the proper combination of hyperparameters to optimize the model performance on the validation set. The complete list of tuned hyperparameters and their candidate values are shown in Table 5.1. The models are trained on powerful graphical processing units (GPU), such as the Nvidia RTX A4000, Nvidia RTX 3060, and Nvidia RTX 4090, to facilitate the complete grid search of hyperparameters by exploiting the high computational capabilities of these GPUs. The selected hyperparameters are bolded in Table 5.1.

The training goal of a neural network is to minimize the loss function. For most binary classification tasks whose output values are processed by a sigmoid function, a cross-entropy loss is typically used, as shown in Eq. (5.4):

$$J = -\frac{1}{N}\sum_{i}[y_i \ln(y_i^p) + (1 - y_i) \ln(1 - y_i^p)] \tag{5.4}$$

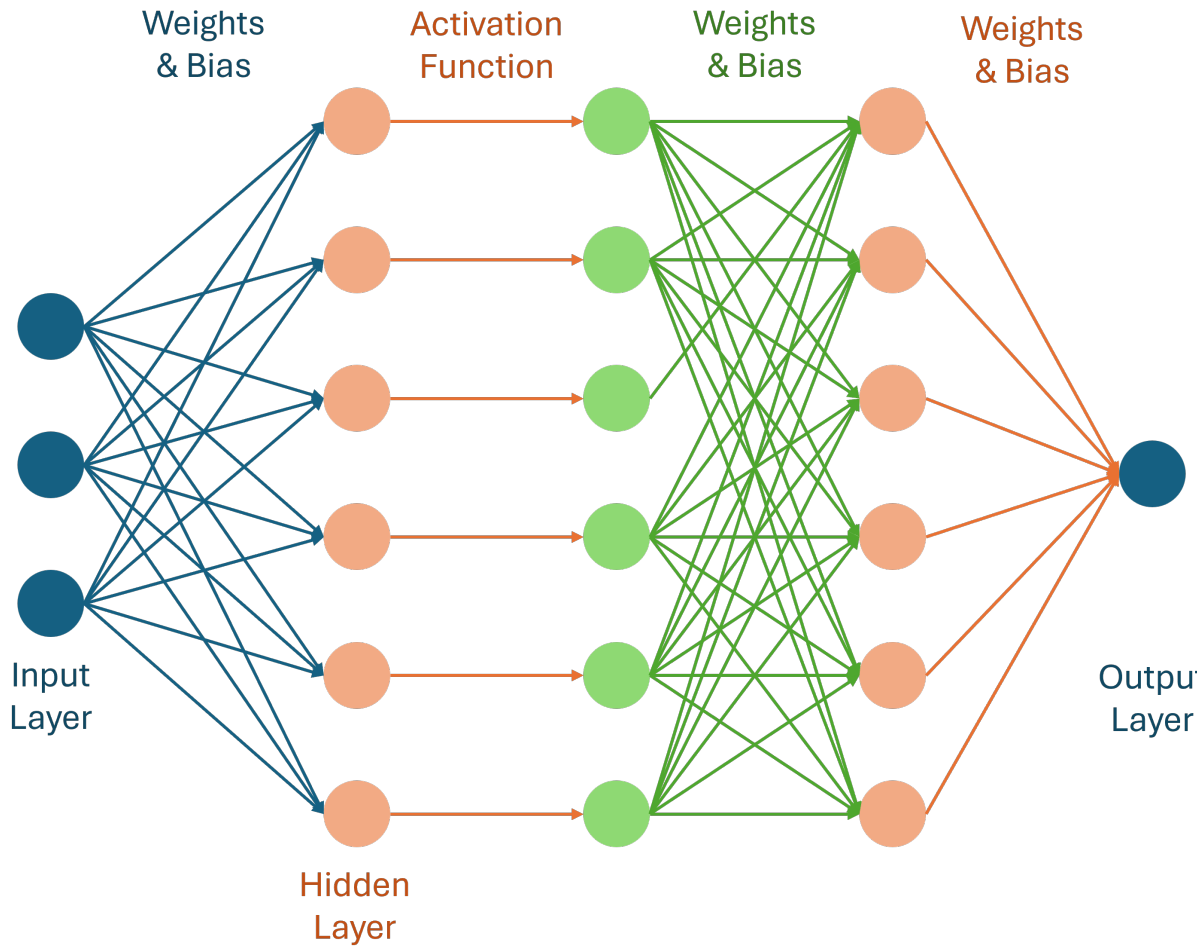


Figure 5.4: The general structure of the Feedforward Neural Network (FNN) is shown here. The hidden layer neurons take a weighted sum from the input layer or the previous hidden layer, which is then activated by nonlinear functions. The output layer takes the weighed sum from the last hidden layer to output the result.

Table 5.1: Classification FNN Hyperparameters and Tuning Range.

Hyperparameters	Candidate Values
Number of Layers	[1,**2**,3]
Number of Neurons	[32,**64**,128]
Learning Rate	[0.005,**0.001**,0.0005,0.0001]
Dropout Rate	[**0**,0.1,0.5]
L2 Regularizer	[**0**,0.0001,0.0005,0.001]

where J is the loss value, y_i is the true value (0 or 1) of the data point i, y_i^p is the predicted value from the model within range [0,1], and N is the number of data points. However, it is important to note that the data is imbalanced because it comes from an industrial toolset that generally runs well with a low but still significant fail rate. All the fail rates for all five datasets are shown in Table 5.2, and most datasets exhibit a fail rate of approximately 2.5%. T5-PM2 is a notable exception with a significantly higher fail rate. This can be partially explained by its small dataset volume, which causes a few FAIL data points to have a large impact on the fail rate, but TM5-PM2's data clearly has more fails than the other tool-module combinations, marking it as different. Thus, by aggregating the TM5-PM2 dataset with other datasets and observing whether model performance increases or decreases, the effects of aggregating less related datasets together can be seen. For all other datasets, a model that always predicts "PASS" will not result in a high loss with the normal cross-entropy loss function, and that behavior will be favored during the training process.

However, such a model is not useful because it will have a 100% false positive rate. In other words, the model will not be able to detect any misprocessed wafers even though it has a very low training loss. To address this issue, a weighted cross entropy algorithm is proposed by multiplying

Table 5.2: Overall Size and Distribution of Datasets.

Dataset	Total Data Points	PASS Data Points	FAIL Data Points	FAIL Rate
T4-PM1	39717	38836	881	2.22%
T7-PM1	23387	23057	330	1.41%
T2-PM2	36335	35461	874	2.41%
T5-PM2	771	667	104	13.49%
T7-PM2	2722	2650	72	2.65%

the weight to data points by the output distribution [106]. The weighted loss function has the form:

$$J = -\frac{1}{N}\sum_{i} w_i[y_i \ln(y_i^p) + (1 - y_i) \ln(1 - y_i^p)] \tag{5.5}$$

where w_i is the weight and N is the total number of data points. The weights for binary classification are calculated as follows:

$$w_i = \begin{cases} \dfrac{N}{n_0} & \text{if } y_i = 0 \\ \dfrac{N}{n_1} & \text{if } y_i = 1 \end{cases} \tag{5.6}$$

where w_i is the i^{th} weight, n_0 is the number of FAIL data points, and n_1 is the number of PASS data points. The weights applied in the loss function emphasizes minority instances to favor models with more comprehensive and balanced performances, thereby mitigating the potential bias that can arise from imbalanced datasets.

The Adam optimizer is applied throughout the entire model training process due to its excellent performance in handling various deep learning tasks [40]. Adam combines the advantages of other optimizers such as AdaGrad and RMSProp to enhance convergence speed and model performance. Model training is conducted over 1000 epochs, with the validation set loss continuously monitored after each epoch. The model with the lowest validation loss up to that epoch is then saved as the best model until the end of the training process. This approach ensures that the model with

best generalization ability is retained, effectively reducing the risk of overfitting and improving the model's generalization capabilities. After the training process, the model is tested on future data of the test set to evaluate its performance.

To further reduce variability in the training and testing process, a cross-validation method is applied. This approach can greatly reduce model variance and improve the reliability of the model performance by averaging the performance of multiple models that are trained on different sections of the original data [39]. First, the modeling dataset is randomly divided into five segments. Then, a segment is chosen as the validation set with the other four segments becoming the training set. By repeating this five times in total, once for each segment, five models are trained. Finally, each model is individually run on the test dataset, and the average of the five scores from the five models is used as the final test score for that dataset. This method enhances the robustness of the model evaluation by ensuring that the performance is consistent across different subsets of the data.

Data Aggregation

Deep learning models, with their complex architectures and numerous parameters, can effectively capture intricate patterns within large datasets, leading to superior performance in most tasks [72]. As a result, deep learning networks have a distinct advantage over traditional machine learning methods, especially when working with training data at the industrial scale. However, industrial datasets often vary significantly in size between different tool-module combinations. For example, in this chapter, as shown in Table 5.2, T7-PM2 and T5-PM2 have dramatically smaller datasets with less than 3000 points compared to T4-PM1, T7-PM1, and T2-PM2, which have more than 20000 points. This variation in dataset size implies that models trained using limited data from a particular tool-module combination for that specific combination will have considerably worse training results compared to tool-module combinations with larger datasets. Additionally, when the validation dataset is limited in size, it can lead to bias in the model, decreasing its ability to generalize because the validation set might no longer accurately represent the general data

distribution. Moreover, even for the three tool-module combinations with large datasets, dataset aggregation will increase the variational and operational coverage of the training data and generally improve model performance. Consequently, maximizing high-quality training data volume is necessary to improve model performance. However, it is not enough to simply merge a large dataset and a small dataset, as that would merely result in a model that predicts the average behavior of the two aggregated datasets. Rather, multiple datasets from various tool-module combinations should be aggregated into a superset of varied data. When trained on this superset, the model performance improves as the larger volume of data better represents the overall process, increases the operational scope, and has more comprehensive information regarding process failures. A large amount of varied training data points that can only be obtained from aggregating data from multiple tool-module combinations refines the model, reduces overfitting, and enhances the robustness of the predictions.

To provide more data to train a model for each tool-module dataset, a data aggregation method is developed and tested that combines multiple datasets, significantly increasing the amount and variety of training data and improving model performance. During this process, the candidate datasets for aggregation must be selected carefully, making the analysis and selection process a critical data processing step. If the chosen tool-module dataset is very different from the current dataset, then the model will likely be misdirected by the new data and fail to retain the distribution of the original dataset. One method to guarantee optimal data aggregation is to exhaustively test all possible dataset combinations, but that is only feasible when there are only a few datasets. The number of possible dataset combinations increases exponentially with the number of datasets; n datasets have $2^n - 1$ unique combinations, making it impractical to exhaustively test all possible combinations when there are many tools and datasets.

To address this issue, this chapter develops an indexing method to evaluate the similarities and differences between datasets. It was then used to select candidate datasets for aggregation, ones that are most similar to the dataset of interest. The indexing method is a point-biserial correlation

analysis, which is a statistical method that measures the relationship between a continuous variable and a binary variable. In this chapter, the point-biserial correlation analysis is conducted on each numerical feature in the dataset with respect to the output binary variables, which provides information about the contribution of each feature to the binary outcome (PASS/FAIL). The analysis involves calculating the correlation coefficient as shown in Eq. (5.7).

$$r_{bis} = \frac{\bar{Y}_1 - \bar{Y}_0}{s_y} \sqrt{\frac{N_0 N_1}{N(N-1)}} \tag{5.7}$$

where r_{bis} is the correlation score between a specific feature and the output, $\bar{Y}_1$ is the average value of the feature for all PASS data points, $\bar{Y}_0$ is the average value of the feature for all FAIL data points, s_y is the standard deviation of the feature for all data points, N_0 is the number of FAIL data points, N_1 is the number of PASS data points, and N is the total number of data points. After the correlation coefficients are calculated for each feature, they are assembled into a characteristic vector. Each dataset has its own characteristic vector, and the difference score between any two datasets is calculated by finding the mean absolute error (MAE) between their characteristic vectors as shown in Eq. (5.8).

$$d = AVG(|\vec{c_1} - \vec{c_2}|) \tag{5.8}$$

where d is the difference score between the two datasets, AVG is an operation that takes the average value of all elements in a vector, $\vec{c_1}$ is the characteristic vector of first dataset, and $\vec{c_2}$ is the characteristic vector of second dataset. After the difference scores are calculated between the current analyzed dataset and all other datasets, the one with the smallest difference score is chosen as the candidate dataset for aggregation. The statistical analysis of the datasets themselves does not require any model training until the candidate datasets for aggregation are selected. This approach significantly reduces the number of models that need to be trained and tested, saving a substantial amount of time and computational resources.

5.2.4 Regression Model

This chapter also explores machine-learning models that quantitatively predict the measured oxide thickness of processed wafers. This model, also called a regression model, is another FNN, though one with a different structure better suited for non-binary outputs. The preprocessed input data for the regression models are the same as that of the classification models, but with the addition of the target oxide thickness as an input feature. This feature is included as the target oxide thickness for each process must be known before the run starts. The model output is the measured oxide thickness, which spans a wide numerical range, from 0 to over 5000 Å, depending on the process and product.

Regression models generally require more data than classification models as the output of the latter is more complex. Thus, this chapter only focuses on training regression models with large datasets [72]. Specifically, T2-PM2, T4-PM1, and T7-PM1 are used to train regression models while T5-PM2 and T7-PM2 are not. The latter two datasets cannot support the training of a complex regression model because the limited data volume can induce problems such as severe overfitting and high variance results. Due to a lack of feasible datasets, data aggregation, which was explored for the classification task, is not conducted for the regression task because there are only three applicable datasets for regression. The available multi-dataset combinations are limited; specifically there are three two-set aggregations and one three-set aggregation. This small sample size means that any conclusions reached through the statistical analysis may be biased. Lastly, regression models do not require any data stratification like classification models because random selection and chronological selection can easily create artificial differences in fail rates between the training and validation sets due to the natural imbalance of the datasets. Because only datasets with substantial data volumes are selected, which effectively reduces the variance in the input features between different time segments, the training-validation split is organized solely by time. Specifically, the first 80% of the total training-validation dataset, sorted chronologically, is used for training, and the remaining 20% is reserved for validation. This approach aims to enhance

model performance by selecting for the regression model that best predicts future data points.

Regression Model Training

The Feedforward Neural Network (FNN) for the oxide thickness regression task applies the ReLU (Rectified Linear Unit) activation function to provide nonlinearity for the model, which is described by the following equation:

$$\text{ReLU}(x) = \max(0, x) \tag{5.9}$$

The ReLU function is chosen because of its promising performance on various training tasks, ability to avoid gradient vanishing problem, and increase in the speed of the training process [93, 98]. In addition, due to how complex it is to train regression models and the need to save computational resources during the hyperparameter grid search, the neural network is designed such that each subsequent hidden layer has half the neurons of the previous layer. This approach helps manage model complexity, reduces the risk of overfitting, and ensures efficient feature extraction. Similar to the classification task case, the tunable hyperparameters and their range of interests are shown in Table 5.3, and the selected hyperparameters are bolded.

Table 5.3: Regression FNN Hyperparameters and Tuning Range.

Hyperparameters	Candidate Values
Number of Layers	[2,**3**]
First hidden layer Neurons	[32,**6**,128]
Learning Rate	[0.0001,**1E-5**,5E-6,2E-6]
Dropout Rate	[**0**,0.1,0.25]
L2 Regularizer	[**0**,0.0001,0.0005,0.001]

A small learning rate and many training epochs (100,000 in this chapter) are essential for training regression models due to the wide range of output values and the highly complex nonlinear

correlations between the 37 input/output features. These conditions ensure that the training process reaches an optimum. Without the small learning rate, the model is sensitive to improper convergence, which causes the training process to jump around the optima or even diverge. By contrast, a small learning rate allows for more precise adjustments to the weights during backpropagation. The extensive number of training epochs ensures that the model has sufficient iterations to learn these complex relationships thoroughly, ultimately leading to better generalization and performance on future data.

Because the output has such a wide range of numerical values, two kinds of loss functions are tested in this chapter: the mean squared error (MSE) and the mean absolute percentage error (MAPE). The MSE loss function is given below:

$$J = \frac{1}{N}\sum_{i}(y_i^p - y_i)^2 \tag{5.10}$$

And the MAPE is described as follows:

$$J = \frac{1}{N}\sum_{i}\frac{|y_i^p - y_i|}{y_i + 1} \tag{5.11}$$

where J is the loss function value, N is the total number of data points, y_i^p is the predicted output value by the regression model, and y_i is the true output value. The plus one term in the MAPE equation prevents the equation from dividing by zero and also avoids ridiculously high loss values when the true output is close to zero. The loss functions are applied to the normalized data scaled by the MinMax Scaler, which keeps small data values that are close to 0 still close to 0 and scales large data values close to 1. As the original data's minimum value is retained, the data normalized by the MinMax scaler also retains the properties of the original dataset, which improves the ultimate performance of the model. When applied, these two loss functions intrinsically favor very different modeling patterns. MSE measures the average of the squares of the errors, treating all

errors equally regardless of the magnitude of the true values. This results in data points with low true values having high percentage errors, as the model may focus more on minimizing absolute errors, which can disproportionately affect smaller values. Conversely, MSE can result in lower percentage errors for data points with high true values, as the absolute errors tend to be relatively smaller in proportion to the larger true values. This can be advantageous when precision for higher value predictions is critical. MAPE, on the other hand, measures the average absolute percentage error, ensuring that the model minimizes the percentage error across all data points. This results in a more uniform percentage error distribution, which is beneficial when the relative accuracy of predictions is more important. Each loss function has its benefits and drawbacks, which is why this chapter uses both to train the model. Using both MSE and MAPE allows for a holistic review of the data and the modeling process. This dual approach provides a more comprehensive understanding of the model's potential across the entire range of output values.

5.3 Results and Analysis

5.3.1 Classification Model Performance

The performance of the classification models proposed in Section 5.2.3 are ideally evaluated within the context of a manufacturing environment. With the classifier model, there are four possible process outcomes: a pass is classified as a pass (true positive), a pass is classified as a fail (false negative), a fail is classified as a fail (true negative), and a fail is classified as a pass (false positive). Of these outcomes, the true positive and true negative outcomes are trivially good outcomes, as the classifier model is correct. The false negative, while not ideal, can be mitigated by manufacturing procedures. If all runs classified as fails are reevaluated at the metrology machine and manually measured to determine whether they truly failed, then the false negatives will be caught and correctly reclassified as passes. Thus, the main manufacturing concern is false positives, as there is no

easy way to identify them; manually measuring all passes in addition to all fails would make the classifier model superfluous.

Generally speaking, most false positives will be eventually caught at future metrology steps or at the final metrology and reliability testing of the end product, which means that they will ultimately have little effect on product quality [19]. The main impact of misidentifying a misprocessed wafer is that, as the misprocessed wafer moves through the process flow, it wastes resources and time. Thus, a high-performing classification model will have a low false positive rate as that minimizes wasted manufacturing resources, and any metrics used to analyze these models must be an indication of their false positive rates.

However, each model does not have a singular, representative false positive rate. Specifically, each model can be tuned to be more aggressive in flagging misprocessed wafers, lowering the false positive rate and increasing the false negative rate, or more conservative, which does the opposite. While confusion matrices are often used to evaluate classifier model performances, they are insufficient to evaluate the overall model performance, as a confusion matrix reflects the results of a single tuning approach. It cannot capture the model's performance across all possible tuning configurations. Other traditional criteria for evaluating classification model performance, such as overall accuracy, are also unsuitable because the binary outputs of all the datasets are highly imbalanced. As previously mentioned, a model that only predicts PASS may have a high accuracy but it will have a 100% false positive rate, rendering it meaningless in actual industrial applications. The Receiver Operating Characteristic (ROC) analysis offers a more robust evaluation method by examining the true positive rate (TPR) and false positive rate (FPR) of the model's predictions on test data at different thresholds. The output from the sigmoid function in the model's last layer is a continuous value in the range of [0,1]. Although a fixed threshold of 0.5 is traditionally used that classifies values higher than 0.5 as pass and those lower as fail, this threshold can be set to any value within the [0,1] range. With different thresholds, different TPR and FPR values are produced. For instance, setting the threshold to 0 (all pass) results in a TPR of 100% and an

FPR of 100%. Conversely, setting the threshold to 1 (all fail) yields a TPR of 0% and an FPR of 0%. Thus, selecting an appropriate threshold involves balancing model sensitivity (TPR) and false acceptance rates (FPR).

In this context, the model's performance is evaluated using the Area Under the Curve (AUC) score. The AUC score is defined as the area under the ROC curve, which plots TPR on the y-axis and FPR on the x-axis across different thresholds. Ideally, a perfect model would achieve a TPR of 100% (max sensitivity) and an FPR of 0% (zero false alarms), resulting in an AUC score of 1. Conversely, a model that makes random guesses would have a TPR of 50% and an FPR of 50%, resulting in an AUC score of 0.5. The ROC-AUC score provides a comprehensive measure of model performance across all possible thresholds, making it particularly useful and widely applied for evaluating models on imbalanced datasets [26].

Single Dataset Model Performances

The model is first trained on single tool-module datasets to evaluate if the training method is effective at making predictions that are significantly better than random guessing, which would have an AUC score of 0.5. For each dataset, five models are created via the cross-validation training process as described in Section 5.2.3. The average AUC score is defined as the mean value of the test scores for each trained model. Note that all ROC graphs are of the model whose performance best matches that of the average of the five models; this is the representative model. The scores for each tool-module combination are shown in Table 5.4. The ROC-AUC plot of the representative model for all five datasets is shown in Fig. 5.5. The single dataset training results in Fig. 5.5 demonstrate that the amount of training data is a key factor in model performance. The two smaller datasets, T5-PM2 and T7-PM2, show unacceptable performance close to near-random guesses (AUC scores of around 0.5). In contrast, the three larger datasets have AUC scores significantly above 0.5, indicating successful model training and effective classification of the PASS/FAIL status of the product. For these three tool-module combinations, the models can achieve a FPR rate of

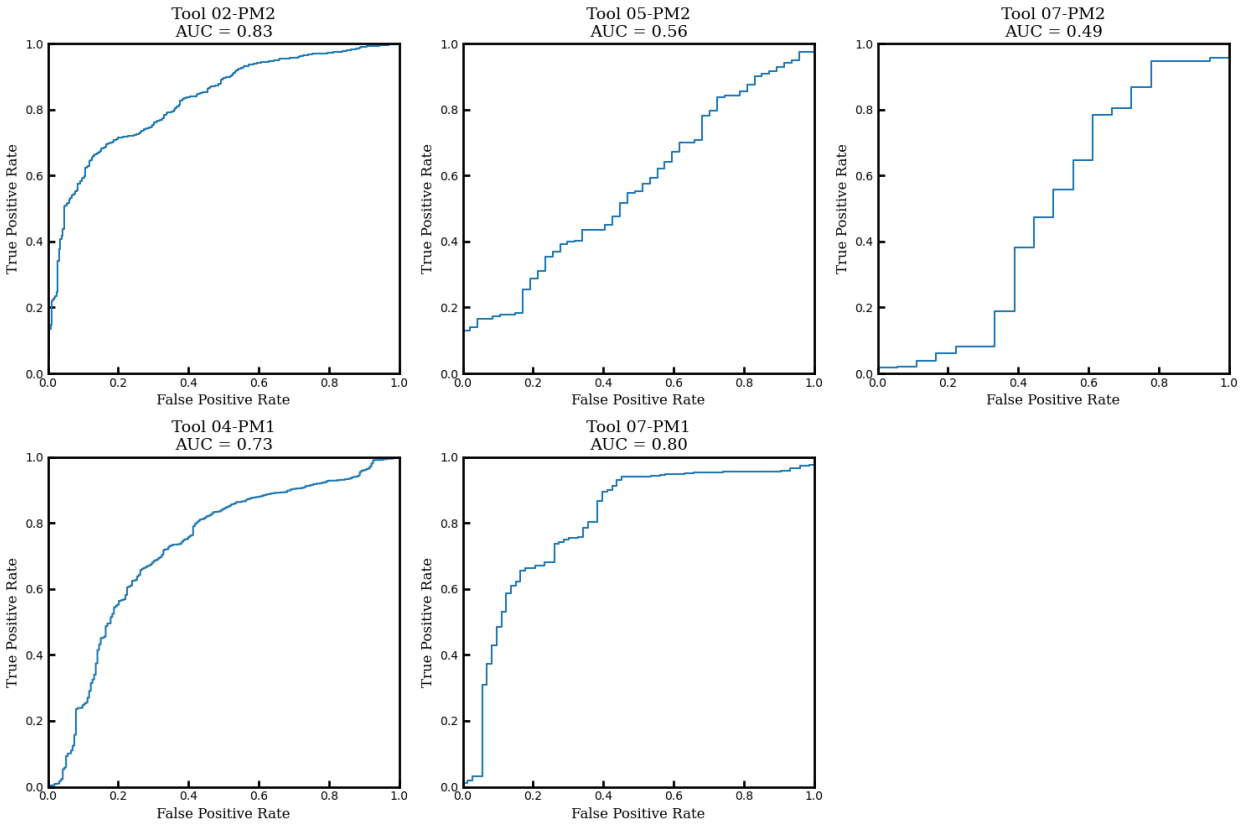


Figure 5.5: ROC plot for all five datasets. The performance of the models trained on datasets T5-PM2 and T7-PM2 is similar to random guesses, while the models based on the other three datasets have superior performances.

Table 5.4: Single Dataset Training AUC Score.

Dataset Name	Average AUC Score
T4-PM1	0.73
T7-PM1	0.73
T2-PM2	0.80
T5-PM2	0.48
T7-PM2	0.52

about 35% to 40% when the TPR is around 80%. This means that, if 10000 wafers are processed with 200 fails and 9800 passes (2% failrate), then 1960 wafers would be false negatives (20% of 9800 passes) and 70 wafers would be false positives (35% of 200 fails). This means that the overall rate of missed fails is less than 1% of the overall product throughput, which is considered high-performing. The single dataset cases suggest that data volume is crucial to model performance; with enough data, the models are able to effectively classify runs between PASS/FAIL. And with even more data, the AUC score can be further improved and the FPR reduced.

Multi Dataset Model Performances

To validate the efficacy of data aggregation in enhancing model performance, the process is first investigated within each module (PM1, PM2). Specifically, this involves forming each unique superset between T4-PM1 and T7-PM1 for PM1 and each superset between T2-PM2, T5-PM2, and T7-PM2 for PM2. This results in three two-set supersets for PM2 and one two-set superset for PM1. PM2 also has the option to aggregate all three datasets. The optimal AUC score for a given tool-module combination is defined as the best AUC score among all the possible supersets that contain that tool-module combination, and the optimal AUC score for each tool-module combination is illustrated in Fig. 5.6. This analysis aims to determine whether combining datasets from different tools within the same module can lead to improved predictive performance, as measured

by the AUC score.

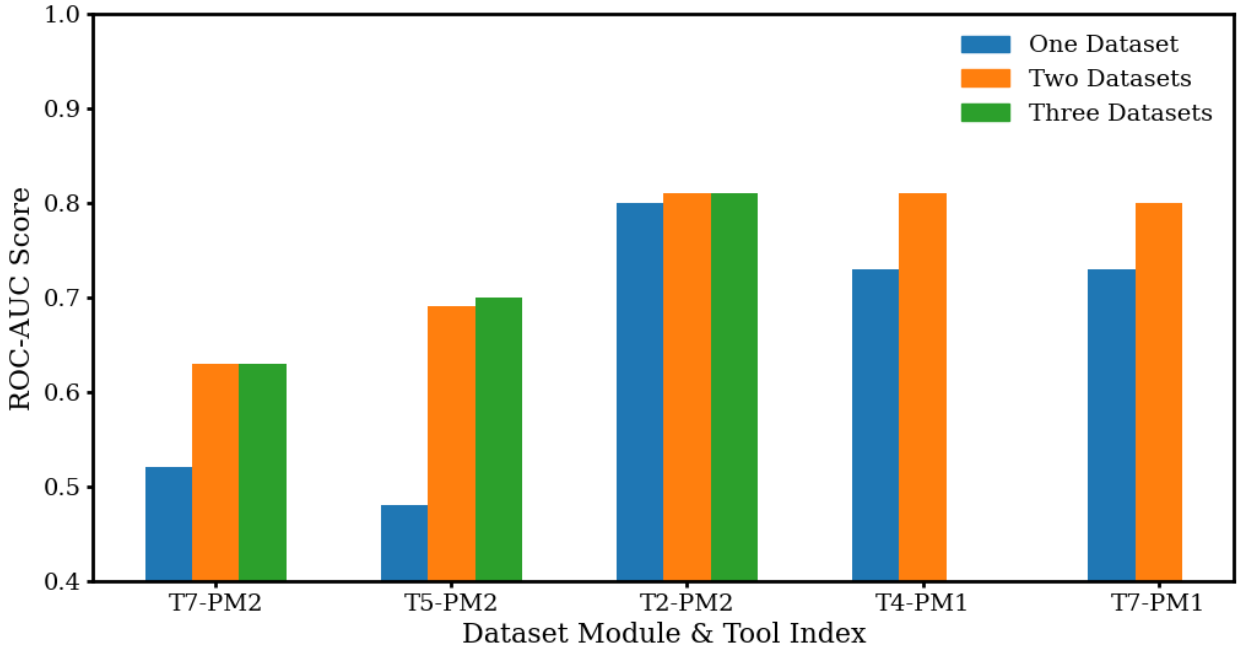


Figure 5.6: Best possible AUC score for all five tool-module combinations. Model performances improve substantially as data increases for all five cases.

From the same-module data aggregation results shown in Fig. 5.6, the AUC score for the T7-PM2 model trained on only the T7-PM2 dataset is just above 0.5, which is an unacceptable performance. However, for the best-case two- or three-set supersets, the AUC score improves past 0.6. The results for T5-PM2 are even better. In the case of T2-PM2, it is clear that the base T2-PM2 dataset contributes the most to the model and that the model receives little benefit when it is aggregated with the other smaller datasets. However, the contribution that the T2-PM2 dataset makes in the orange and green bars of the T7-PM2 and T5-PM2 models demonstrates substantial performance improvement for Tools 05 and 07, which have smaller datasets. Generally, data aggregation substantially improves model performance for tool-module combinations with smaller datasets, such as T5-PM2. Furthermore, the three-set aggregation in PM2 allows the models to achieve a FPR of around 50% with a TPR of about 80%, as shown in the ROC plot in Fig. 5.7. For the base model trained on only one dataset, an 80% TPR would correspond to an 80% FPR,

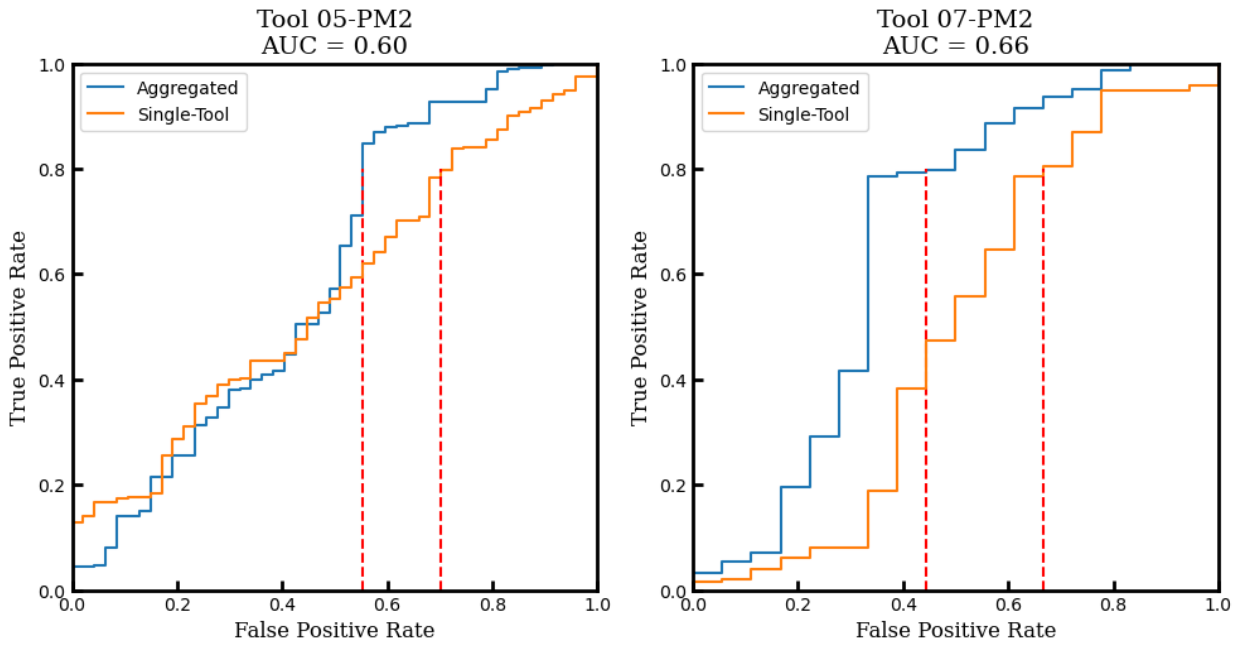


Figure 5.7: ROC plots of the representative models of T5-PM2 and T7-PM1, which were trained by aggregating all available datasets in the module. The performances are noticeably better than that of random guessing and single-set models. The FPR values for an 80% TPR value have also significantly improved.

which would result in an overall missed fail rate of 1.6% given a 2% fail rate. However, the model trained with multi-dataset aggregation would have a missed fail rate of 1%, a significant improvement. Additionally, even the tool-module combinations that have the largest datasets improve with data aggregation. Compared to their single-set results in Fig. 5.6, the AUC scores of the aggregated models reach over 0.8 for both T2-PM2 and T7-PM1, as shown in Fig. 5.6. These results underscore the effectiveness of data aggregation in bolstering model accuracy, especially for tool-module combinations with limited datasets.

Candidate Dataset Selection Method

While the previous section shows that data aggregation improves model performance, it does not explain how it should be conducted. Additionally, there is no easy way to evaluate a model's performance without actually developing and testing the model. Even in Fig. 5.6, the displayed

datasets had varying levels of improvement. Thus, to minimize the presence of false positives, it is necessary to examine how to optimally aggregate datasets together. To that end, five datasets (T4-PM1, T7-PM1, T2-PM2, T5-PM2, T7-PM2) are examined to assess the effectiveness of the indexing method explained in Section 5.2.3 that is used to select ideal datasets to aggregate with the base dataset, also called candidate datasets. The difference scores for each dataset pair as calculated with Eq. (5.8) are shown in Table 5.5:

Table 5.5: Difference score between each pair of datasets.

	T4-PM1	T7-PM1	T2-PM2	T5-PM2	T7-PM2
T4-PM1	N/A	0.019	0.017	0.083	0.037
T7-PM1	0.019	N/A	0.011	0.083	0.032
T2-PM2	0.017	0.011	N/A	0.080	0.029
T5-PM2	0.083	0.083	0.080	N/A	0.101
T7-PM2	0.037	0.032	0.029	0.101	N/A

The candidate dataset selection criterion for a given dataset is to choose the paired dataset with the smallest difference score. For example, to improve the T4-PM1 model, the T4-PM1 dataset should be aggregated with the T2-PM2 dataset, as the T4-PM1/T2-PM2 pair has the smallest difference score (0.017) compared to the T4-PM1/T7-PM1 (0.019), T4-PM1/T5-PM2 (0.083), and T4-PM1/T7-PM2 (0.037) pairs. For the same reason, the candidate datasets for T5-PM2 and T7-PM2 are both T2-PM2; the T5-PM2/T2-PM2 (0.080) pair is the smallest amongst all the T5-PM2 pairings, and the same holds for the T7-PM2/T2-PM2 pair. Note that the candidate dataset relationship is not necessarily true in reverse. While the candidate dataset for T5-PM2 may be T2-PM2, the candidate dataset for T2-PM2 is not necessarily T5-PM2. From Table 5.5, it can be seen that the candidate dataset for T2-PM2 is actually T7-PM1, with a difference score of 0.011. Additionally, to aggregate three datasets, or when choosing the second candidate dataset, the difference score has to be recalculated between the current two-set superset and all the other datasets. For example,

to determine the candidate dataset for the T4-PM1/T2-PM2 superset (SS1), the difference scores for the SS1/T7-PM1, SS1/T5-PM2, and SS1/T7-PM2 pairs must be recalculated.

To further assess the effectiveness of this data aggregation metric, four datasets are examined in a case study: all three PM2 datasets and T4-PM1. It is still possible to exhaustively test the 15 possible unique supersets formed from these four datasets, which will allow us to evaluate both the idea that data aggregation generally improves model performance and the effectiveness of the candidate dataset selection method. First, the best possible AUC score for each tool-module combination is found by exhaustively creating a model for every possible superset and then selecting the superset that yields the highest AUC score for that tool-module. The best possible AUC score at each superset size is shown in Fig. 5.8. Then, the best possible AUC score is compared to the AUC score obtained by aggregating candidate datasets, and this is shown in Fig. 5.9 for two-set supersets and Fig. 5.10 for three-set supersets.

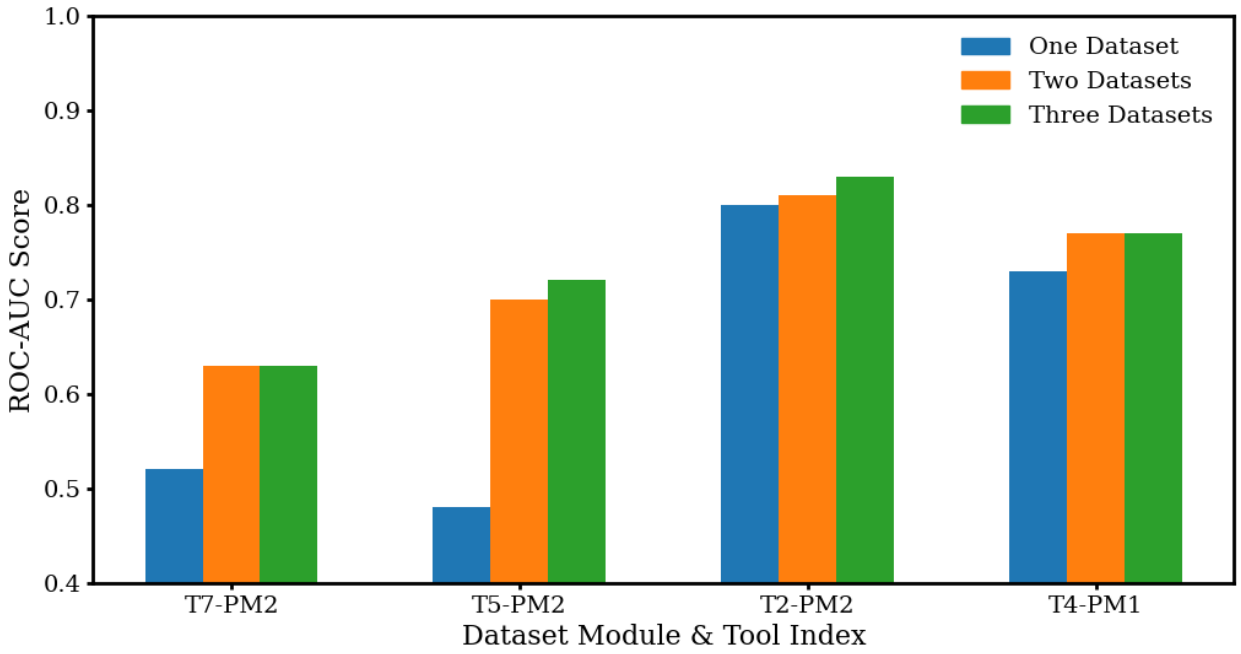


Figure 5.8: Best possible AUC score as a function of how many datasets are aggregated among the four datasets. Model performance improves significantly for all tool-modules as data aggregation increases.

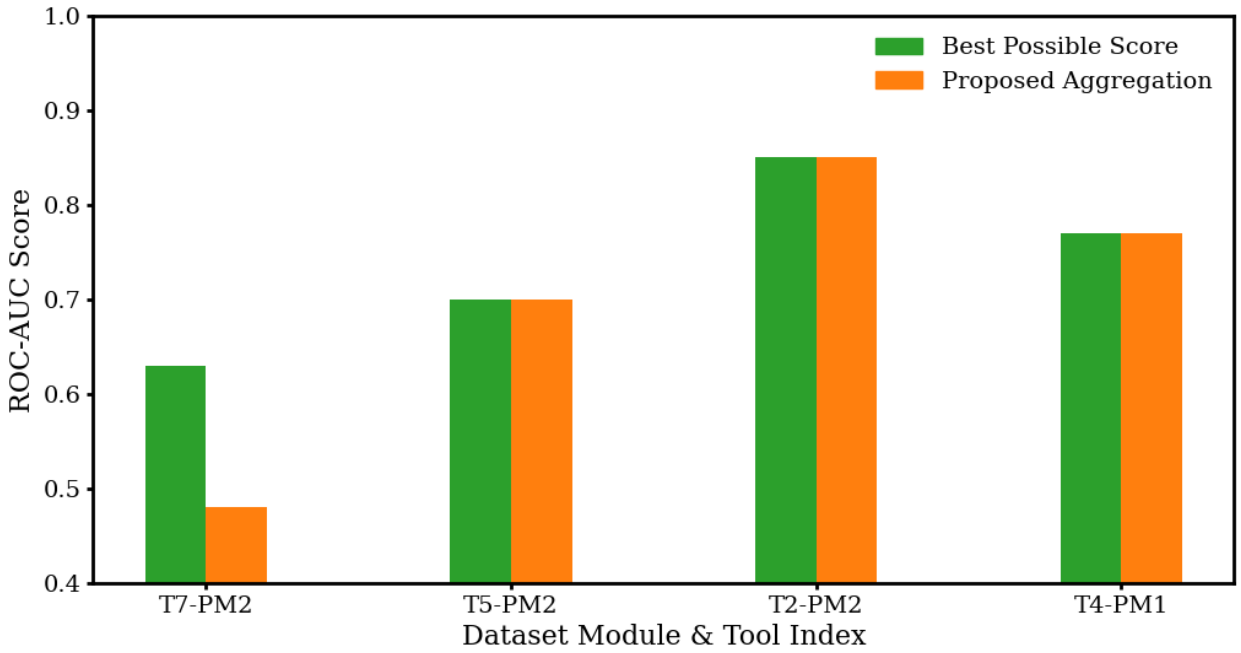


Figure 5.9: Comparison between the best AUC score for a **two-set** superset and the AUC score for the superset proposed by the candidate dataset selection method. Proposed AUC scores conform with the best-case scores for most tool-modules.

Fig. 5.8 validates the idea that data aggregation generally improves model performance because the best possible AUC scores for all four tool-module combinations increase as more datasets are aggregated. Additionally, in Fig. 5.9 for two-set aggregation and Fig. 5.10 for three-set aggregation, the AUC scores of the proposed data aggregation strategy based on statistical analysis methods aligns with the best possible score, with T7-PM2 being an exception in both cases. The exception of T7-PM2 can be explained by examining the difference score between its historical data and its future data. In Table 5.6, the difference score between the historical data used to train models and the future data used to test models is displayed for five tool-module combinations. Notably, the difference score for T7-PM2 is the largest of the examined datasets, which implies that there is significant variability between the modeling set and testing set that may impact data aggregation effectiveness as the data aggregation strategy is purely based on the historical dataset. This indicates that the proposed statistical analysis methods are effective for this case study and

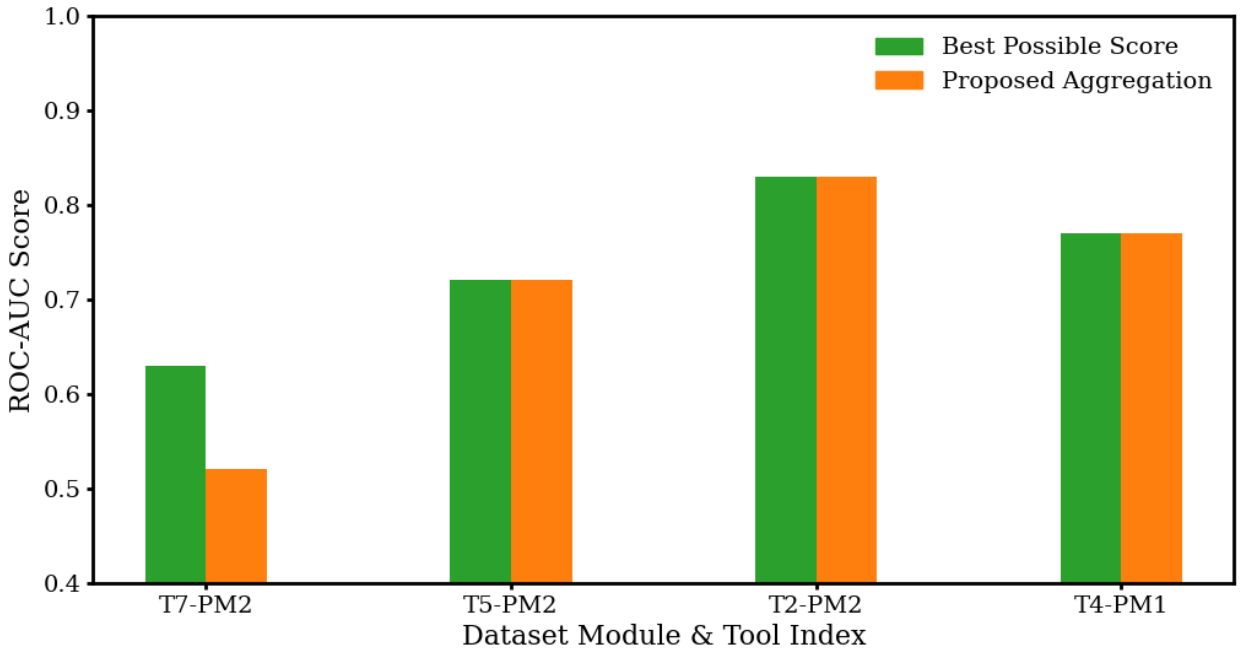


Figure 5.10: Comparison between the best AUC score for a **three-set** superset and the AUC score for the superset proposed by the candidate dataset selection method. Proposed AUC scores conform with the best-case scores for most tool-modules.

can be considered as a preliminary solution for selecting candidate datasets for aggregation.

Table 5.6: Historical and Future Data Difference Score.

Dataset Name	Difference Score
T4-PM1	0.089
T7-PM1	0.052
T2-PM2	0.082
T5-PM2	0.090
T7-PM2	0.093

One of the most important applications of the candidate dataset selection method is to address scenarios where numerous tool-module combinations are involved. In these situations, it is impractical to exhaustively test all possible combinations of data aggregation to determine the ideal superset for each tool-module combination. To further test the capabilities of the candidate dataset

selection method, the five datasets in Table 5.5 are reexamined. For this analysis, not all possible superset combinations are tested. Instead, only the supersets proposed by the candidate dataset selection method are examined. The resulting AUC scores are then evaluated to validate the effectiveness of the proposed aggregation method that was previously demonstrated to work well for the four dataset case study. This approach aims to show that the aggregation strategy will continue to be feasible even as the number of datasets increases and prove that statistical analysis methods can identify the ideal candidate dataset without training numerous models. Among the five datasets, the candidate dataset for T2-PM2 is T7-PM1 as this pair has the lowest difference score (0.011), and the remaining datasets have the same candidate datasets as in the previous case study. The aggregation AUC scores are shown in Fig. 5.11, and a specific example is shown in Fig. 5.12.

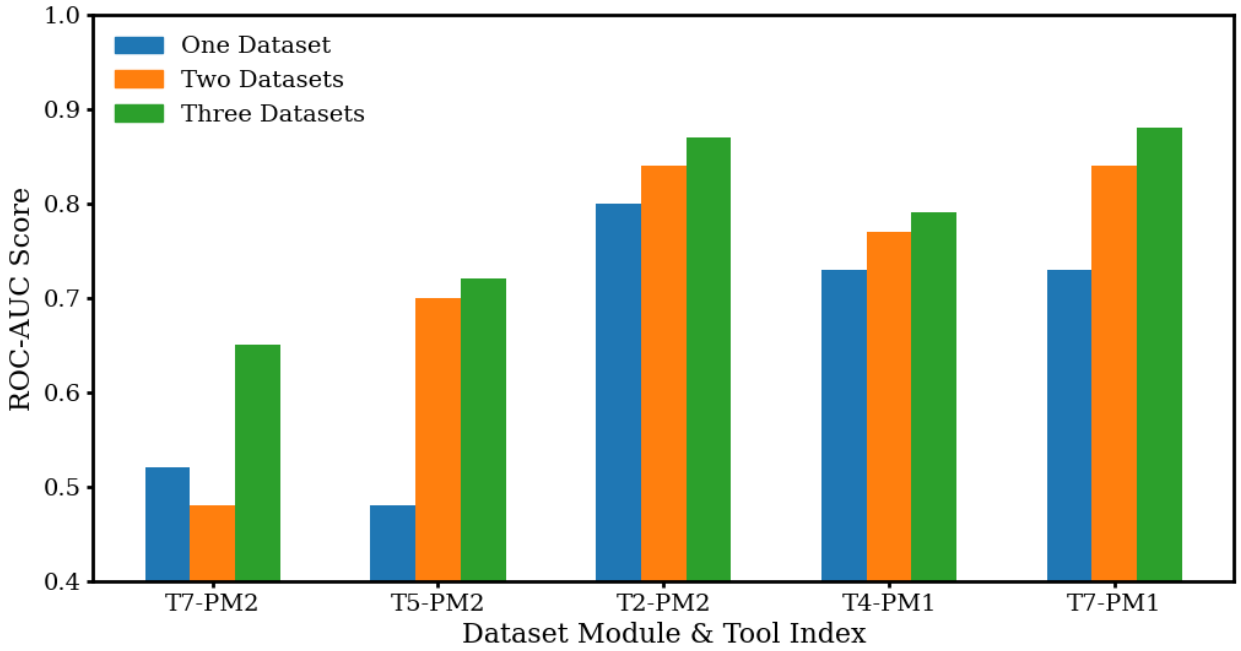


Figure 5.11: AUC scores for the models trained on the supersets proposed by the candidate dataset selection method. The AUC scores for all the tool-module combinations improve as more datasets are aggregated into the modeling set.

Fig. 5.11 illustrates a noticeable improvement in the performance for all the tool-module models. Specifically, T2-PM2 and T7-PM1 achieve AUC scores of 0.87 and 0.88, respectively, which

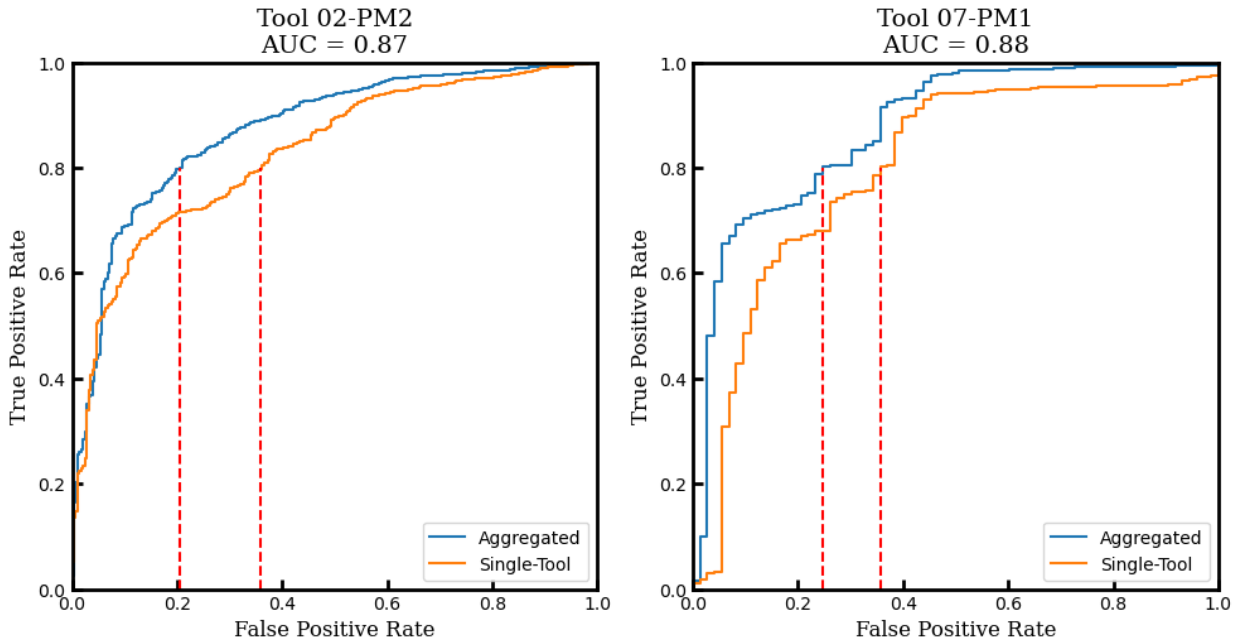


Figure 5.12: ROC plots for model performance on T2-PM2 and T7-PM1, which are trained by aggregating three datasets. The performances are noticeably better than the models trained on single datasets. The FPR values at 80% TPR are improved from good (around 40%) to perfect (around 20%).

is considered high performing. As shown in Fig. 5.12, the ROC plots for T2-PM2 and T7-PM1 indicate that aggregating data has lowered the FPR from between 35%-40% to 20% at 80% TPR, representing a significant improvement.

Fig. 5.11 also demonstrates the importance of data volume and how data aggregation can supplement datasets with small volumes. This is especially in the case of T7-PM2, which previously underperformed in the four dataset case study. Table 5.2 states that it has a small volume of data of 2722 compared to T4-PM1, T7-PM1, and T2-PM2, which have dataset sizes in the tens of thousands. For the cases where only one or two datasets are used, the T7-PM2 model performs poorly, with an average ROC-AUC score of 0.5, which is the same as randomly guessing. However, by aggregating three datasets, it achieved an AUC score close to 0.7 and an FPR of about 40% with a TPR of around 80%. This demonstrates the effectiveness of the data aggregation strategy based on statistical analysis methods at enhancing model performance, particularly in scenarios where there are so many datasets that it is impractical to exhaustively test all possible combinations.

While data aggregation is certainly powerful, it also has its own limitations. The two tool-module combinations with the smallest datasets, T5-PM2 and T7-PM2, cannot attain a FPR of 20% at a TPR of 80% like T2-PM2, a tool-module with a large dataset. This is because the original dataset is simply too limited. Although data aggregation can greatly increase the amount of training data by combining similar datasets, the newly added data will never perfectly follow the distribution of the original data, or in other words, data aggregation dilutes the "identity" of the original dataset. This forms a complex balance between data aggregation increasing the data volume, which improves model performance, and data aggregation diluting the identity of the original dataset, which decreases model performance. Additionally, the improvement effect from increased data volume experiences diminishing returns as seen in Fig. 5.11; T7-PM1's performance experiences a sizable increase when moving from one to two datasets, but its performance only experiences a minor boost when moving from two to three datasets. Thus, the diminishing returns of increased data volume and the cost of identity dilution implies that there exists an optimum

where further data aggregation would lead to decreased rather than increased model performance.

This optimum will also differ for each individual classification problem and dataset. For example, complex problems naturally require larger datasets in comparison to simpler problems, which reduces the diminishing returns effect of increased data volume. Additionally, the dilution extent of the original dataset's identity is dependent on the potential candidate datasets; if the datasets to be aggregated are very similar, then the dilution effect will be weak whereas if the datasets to be aggregated are very different, the dilution effect will be strong. Nonetheless, the improved AUC scores and reduced FPRs highlight the efficacy of data aggregation at limiting the number of missed misprocessed wafers. A key part to data aggregation is the candidate dataset method and its ability to search for ideal datasets that optimize model accuracy and efficiency when dealing with extensive datasets as randomly selecting datasets will result in minor improvements if not decreases in performance. It should be mentioned that the point-biserial analysis and difference scores are not guaranteed to nominate the ideal candidate dataset. Nevertheless, the results of this chapter demonstrate that the proposed data aggregation method can still significantly improve the performance of industrial classification models, validating the practicality and effectiveness of the approach in real-world applications.

5.3.2 Regression Model Performance

The regression model is evaluated with the median percentage error metric. Percentage error is chosen as it is commonly used in industrial settings for numerical data that spans a wide range from 0 to over 5000 Å, and it is calculated with the following equation:

$$J_{percentage} = \frac{|y_{pred} - y_{true}|}{y_{true}} \tag{5.12}$$

where $J_{percentage}$ is the percentage error, y_{pred} is the predicted oxide thickness, and y_{true} is the measured oxide thickness. y_{true} spans from 0 to 5000 Å, and some values are close to 0 Å, e.g.

0.001 Å. The percentage error for these data points with small y_{true} values will be abnormally large despite their small absolute error. To minimize the influence of these outliers, the median, rather than the mean, is used to represent the overall percentage error as it is a more robust and representative measure of model performance. The R^2 criterion is not applied here because of the imbalanced data distribution; the data points are dense around a low target region (from 0 to 200) and sparse at a high target region (from 200 to over 5000). A model that mostly focuses on the dense, low target region can still receive a good R^2 score even if it performs badly on the sparse, high target region. Thus, the median percentage error is a better criterion for evaluating the overall model performance on all ranges of data. As previously stated, the regression task is more complex than the classification task, so only the T2-PM2, T4-PM1, and T7-PM1 datasets are analyzed. Furthermore, since the training, validation, and test datasets for the regression task are all sorted by time rather than randomly selected as was the case for the classification task, the median percentage error of all three datasets is an informative metric and will be shown.

Since the target oxide thickness is always known, a benchmark model can be constructed. Specifically, the benchmark model is defined as a model that always predicts the measured oxide thickness to be the same as the target oxide thickness; it always guesses that the product will pass metrology. Then, by calculating the median percentage error of this benchmark model on the validation set, it can act as a baseline for performance evaluation. The minimum requirement for an acceptable regression model is to outperform the benchmark model. By comparing the performance of various regression models against this benchmark, we can better assess the regression model's accuracy at predicting the measured oxide thickness, especially in an industrial context where reliability is crucial.

MAPE Training Results

The results for all the models trained with the Mean Absolute Percentage Error (MAPE) loss function for the three specified datasets split by the training, validation and test datasets are shown

is shown in Fig. 5.13. Fig. 5.13 shows that, although the trained regression model outperforms the

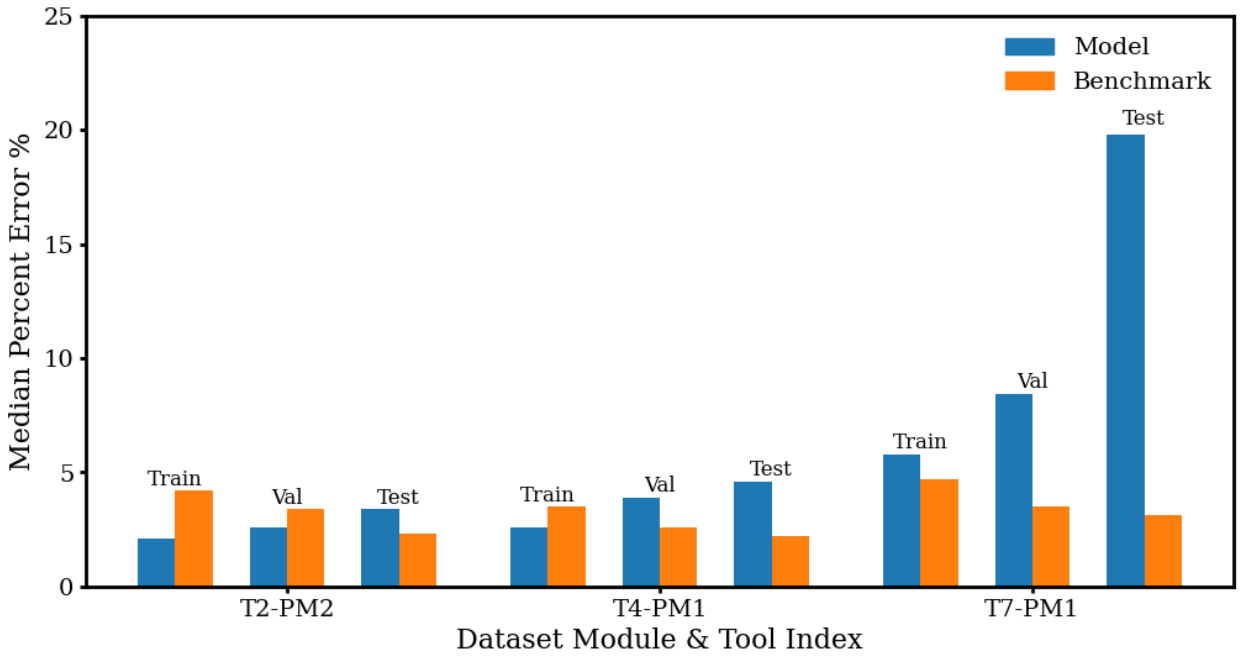


Figure 5.13: The median percentage error after training with the **MAPE** loss function, compared to the benchmark model. Each tool-module labeled on the x-axis has three groups of bars, which represent the training, validation and test sets, respectively. Within each groups of bars, the blue bar is the median percentage error of the trained regression model, and the orange bar is the median percentage error of the benchmark model. Fig. 5.14 to Fig. 5.18 have the same formatting.

benchmark model on the training and validation datasets for T2-PM2, the regression model still underperforms in comparison to the benchmark model on the test set. For T4-PM1, the trained regression model even fails to outperform the benchmark model on the validation set. Moreover, for T7-PM1, the trained model does not outperform the benchmark model on any of the dataset. Overall, it is evident that any regression models trained with the MAPE loss function cannot surpass the benchmark model when evaluated over the entire dataset.

Even when the data is divided into runs with a target oxide thickness value lower than 200 Å (low target) and runs with a target oxide thickness higher than 200 Å (high target) as shown in Fig. 5.14 and Fig. 5.15, the previous conclusion remains consistent. The regression model trained with the MAPE loss function does not outperform the benchmark model on either the low target or the

high target test sets. The designation of "low" and "high" targets is arbitrarily determined by the real-world behavior of the tool-module combinations. Specifically, the runs with a target below 200 Å and those above 200 Å have very different behaviors. By splitting up the data and analyzing it in detail, it gives a more detailed and comprehensive sense of the overall model performance by examining the model performance in different applications.

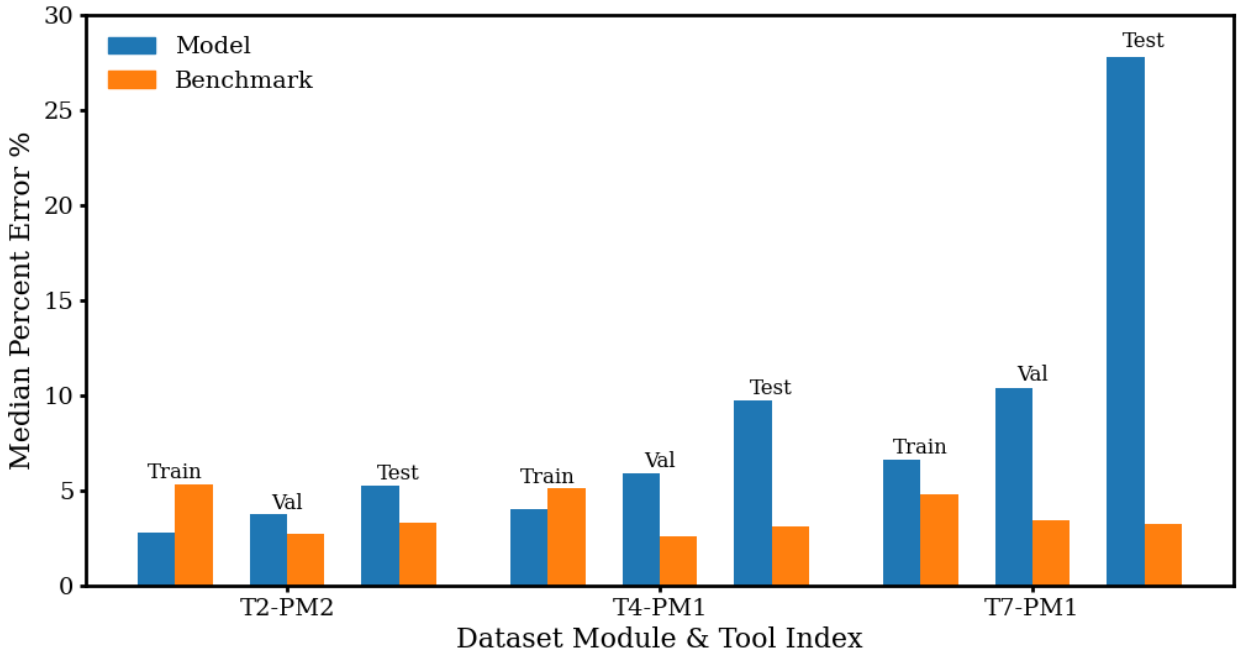


Figure 5.14: Comparison of the median percentage error between various regression models trained with the **MAPE** loss function for the training, validation and test sets and the benchmark model on runs with target oxide thicknesses of **less than** 200 Å.

MSE Training Results

The results for all the models trained with the Mean Square Error (MSE) loss function for the three specified datasets split by the training, validation and test datasets are shown in Fig. 5.16. Fig. 5.16 shows that the median percentage error of the regression model trained with the MSE loss function is worse than that of the regression model trained with the MAPE loss function. This is expected as the MAPE loss better aligns with the evaluation criterion. The regression models

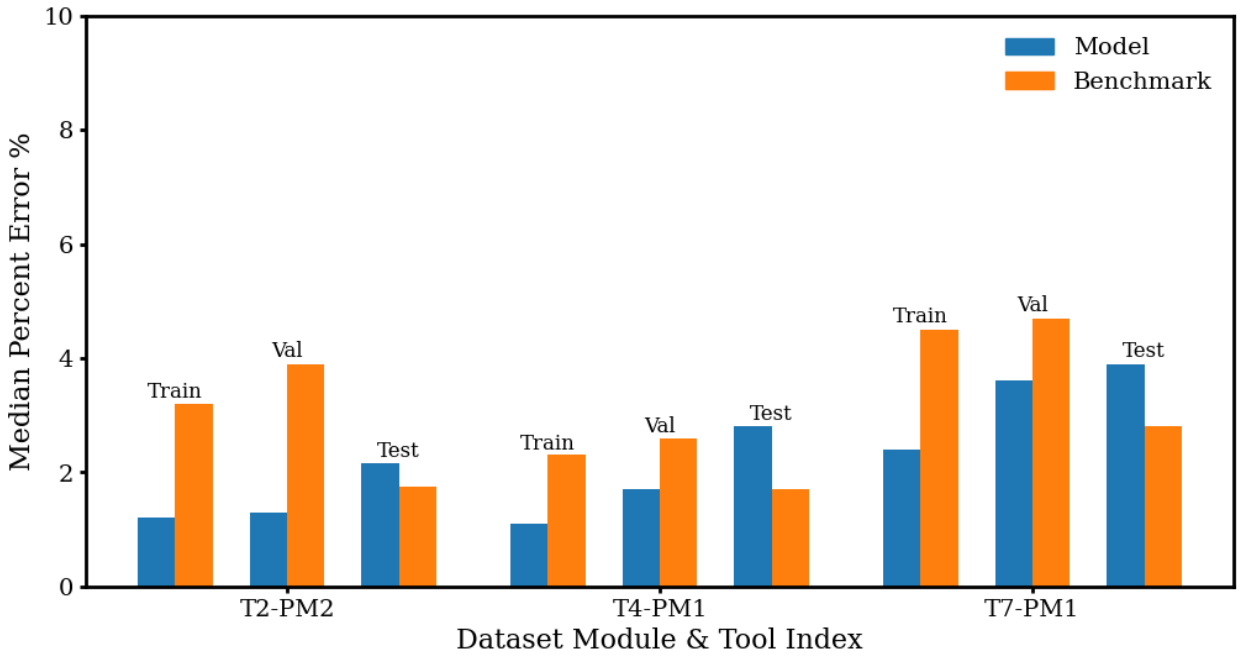


Figure 5.15: Comparison of the median percentage error between various regression models trained with the **MAPE** loss function for the training, validation and test sets and the benchmark model on runs with target oxide thickess of **over** 200 Å.

trained with the MSE loss function perform worse than the benchmark model for all three tool-module combinations across the training, validation, and test sets. This trend is also evident in the model's performance on low target data points, as shown in Fig. 5.17. Due to the intrinsic properties of the MSE loss function, which treats absolute errors on the high and low target data points the same, the median percentage error on runs with low targets is even higher than the median percentage error on all runs. Conversely, this means that the model performs exceptionally well on runs with high targets, as shown in Fig. 5.18. When only examining the runs with high targets, the median percentage error of the trained regression models is significantly lower than those of the benchmark model across the training, validation, and test datasets.

The results shown in Figs. 5.16 to 5.18 prove that the MSE loss function can achieve exceptional performance in terms of median percentage error for runs with high targets. Nevertheless, if the runs with high targets only constitute a negligible portion of the overall dataset, then the good

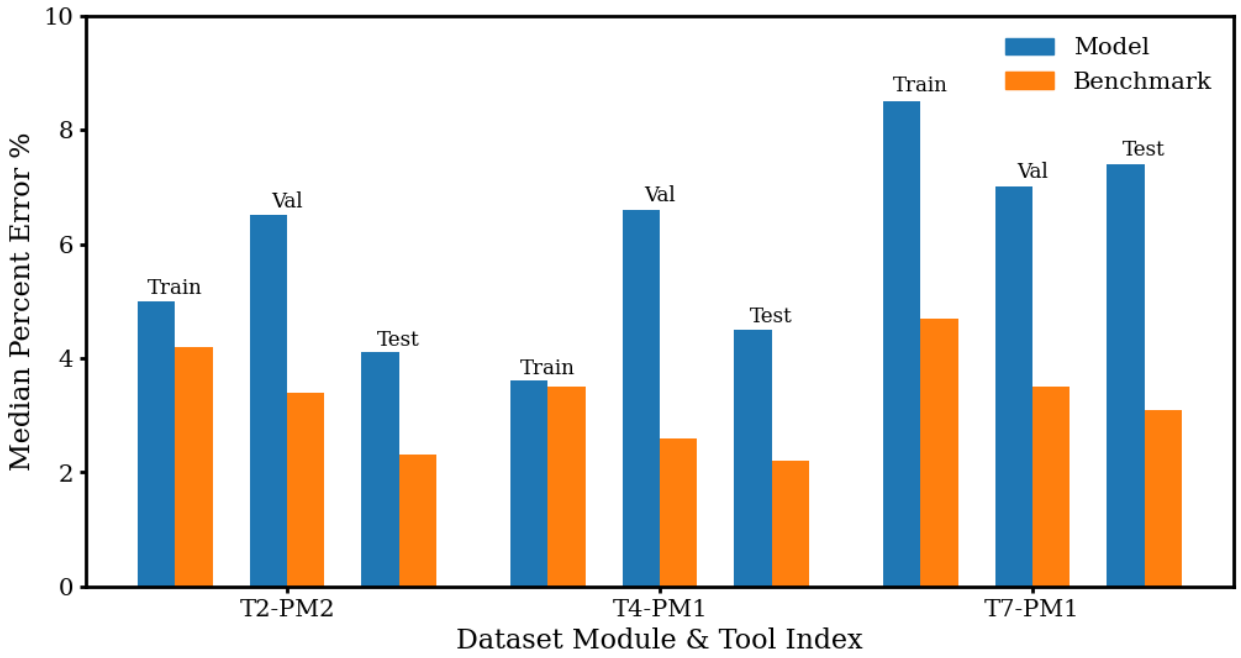


Figure 5.16: The median percentage error after training with the **MSE** loss function, compared to the benchmark model. For all tool-module combinations and all datasets, the trained regression model performance is worse than that of the benchmark model.

performance of the regression model in this category is not particularly noteworthy. In this chapter, however, the high target data points contribute significantly to the overall dataset, as shown in Table 5.7. For instance, over one-third of the data points in the test sets of T2-PM2 and T4-PM1

Table 5.7: Percentage of runs with High Targets in each dataset.

Tool-Module	Training	Validation	Testing
T2-PM2	20%	17%	38%
T4-PM1	21%	39%	38%
T7-PM1	8%	18%	11%

have high target oxide thicknesses.

Given this substantial representation, although the model trained with the MSE loss function cannot be confidently used in all cases, it holds a distinct advantage in predicting the measured oxide thickness when the target oxide thickness is high. Since the target value is always known

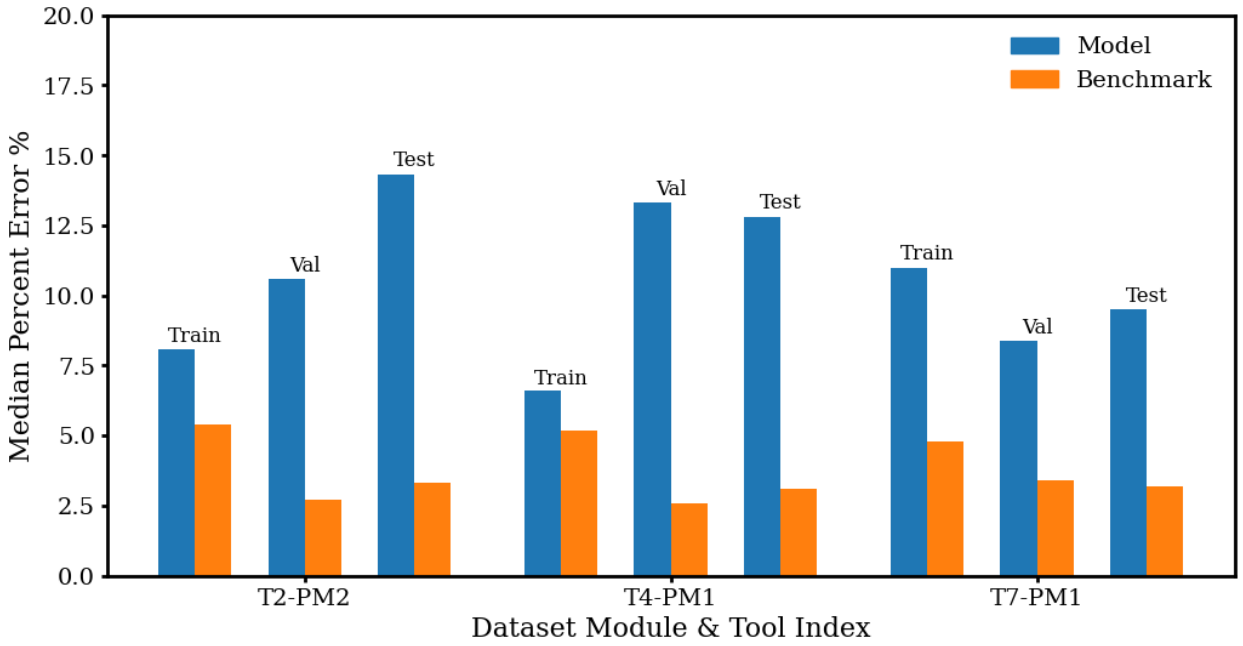


Figure 5.17: Comparison of the median percentage error between various regression models trained with the **MSE** loss function for the training, validation and test sets and the benchmark model on runs with **low** target oxide thicknesses. The performance of the regression model here is even worse than in Fig. 5.16.

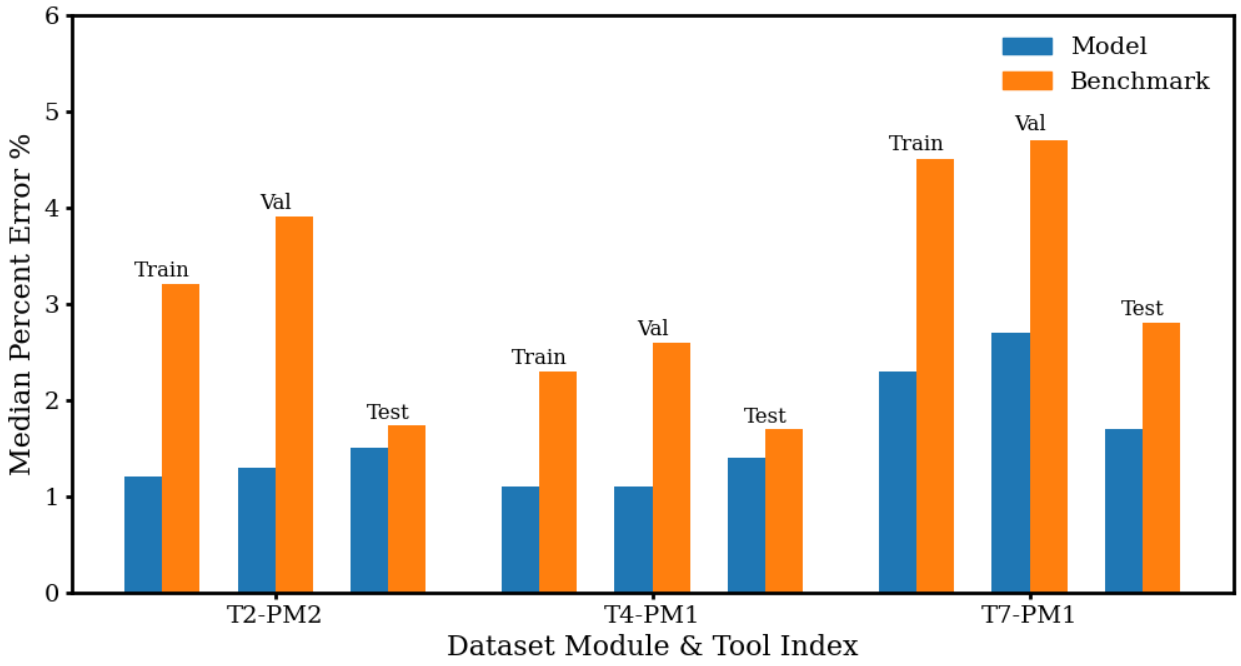


Figure 5.18: Comparison of the median percentage error between various regression models trained with the **MSE** loss function for the training, validation and test sets and the benchmark model on runs with **high** target oxide thicknesses. The performance of the regression model is better than that of the benchmark model on all examined runs for all three training, validation and test datasets.

before processing, this characteristic can be strategically utilized to employ the model in areas where it shows strong performances.

The poor performance of the regression models for both the MSE and MAPE loss functions, compared to that of the benchmark model, can primarily be attributed to the exceptionally high pass rate, which reaches 98% in some datasets. In industry, a common pass criterion is for the result to be within a specified threshold of the target value. This high pass rate indicates that most runs are successfully etched and that their measured oxide thickness is very close to the target thickness, resulting in the benchmark model having a very low median percentage error in most cases. However, this means that it is very difficult to train a regression model that outperforms the benchmark model across all target values. This stands in stark contrast to the results of the classification model, where the favorable ROC-AUC curves in Fig. 5.12 show that the trained models can and do outperform a benchmark model that randomly guesses the outcome.

The reason for the different results stems from the difference in complexity between the tasks. The available process data contains enough information to train a high-performing classification model that simply allocates the results into two classes. However, the regression model must predict the exact oxide thickness value from a wide range of possible values. As the regression task is many times more complex than the classification task, a high-performing regression model requires process data that contains deep insight into the process itself. Thus, the reason why the regression models do not outperform the benchmark model is because the process data does not contain enough information regarding the process. To improve the performance of the regression models, more complex process data, such as time-series data, must be incorporated into the training process. In conclusion, machine learning-based soft sensors are powerful at predicting the physical properties of the process, but only if the model is trained on sufficiently insightful process data collected from physical sensors.

5.4 Conclusion

This chapter describes machine learning-based soft sensors trained on process data from five industrial etching reactors for two tasks: PASS/FAIL classification and oxide thickness regression. A data aggregation method is proposed to improve the model performance for the predictive classification task. In addition, for the regression task of predicting measured oxide thickness, both MSE and MAPE losses are tested for model training. The results presented in this chapter validate the hypothesis that data aggregation and statistical analyses can significantly enhance the binary PASS/FAIL prediction accuracy and robustness of feed forward neural network models on industrial etching reactors. This is done by aggregating datasets using point-biserial correlations and difference scores as metrics to choose candidate datasets for aggregation, particularly for tools with initially small data volumes. The two datasets with the smaller number of data points improved from a score barely above random guessing ($AUC \approx 0.5$) when trained on a single dataset to a significantly better performance ($AUC \approx 0.65$) with a 40% false positive rate (FPR) at a 80% true positive rate (TPR) when trained on an aggregation of three datasets out of five possible datasets. The other datasets with relatively larger datasets still benefited from data aggregation; their model performance improved from $AUC \approx 0.8$ to $AUC \approx 0.9$, and the FPR improved from around 35% at 80% TPR to to 20%. The statistical analyses accurately chose the best candidate dataset for aggregation, aligning with the results shown by exhaustively testing every possible dataset combination. These results confidently show the effectiveness of data aggregation and using statistical methods as an aggregation strategy. Additionally, for regression models that predict oxide thicknesses, while the MAPE loss function was more effective for overall percentage error minimization, the trained model could not outperform the benchmark model that always predicted the measured oxide thickness to be the target thickness. Although the model trained with the MSE loss function did not yield satisfactory predictions across all data points, it exhibited exceptional performance for runs with high target oxide thickness with target values of over 200 Å. This finding

is useful for specific processes with high target values, as these target values are typically known in industrial settings. Consequently, the strategic use of MSE for processes with high target values can enhance regression model performance in these specific scenarios.

Chapter 6

Conclusion

This dissertation discussed the challenges and emerging solutions faced in the semiconductor fabrication industry with a particular focus on the atomic layer etching process. In manufacturing environments, there is a wealth of process data that can be used to train powerful prediction models and control systems. However, the infrastructure and methods needed to effectively implement these cutting edge manufacturing techniques is missing. Thus, this dissertation uses first-principles multiscale modeling to produce process data and simulate the impacts and effects that data-driven modeling can bring to the semiconductor manufacturing industry. This effort was successful as the data-driven models were able to effectively learn the information contained within the first-principles models and predict future states more quickly. This increased simulation speed allowed data-driven models to be used to predict key process variables that are classically difficult to measure, such as kinetic reaction rates. Even with noisy industrial data, as long as proper care is taken to clean and correct the data, highly effective data-driven models can be established. Finally, these data-driven models can be made more robust by aggregating process data from different sources to improve the models' ability to adapt to disturbances and industrial noise.

Key contributions were made toward the usage of data-driven models in the industrial manufacturing space. Specifically, data aggregation methods for improving data-driven models' perfor-

mances were proposed and have since been adopted within Seagate. Future work on the research subject presented in this dissertation includes adjusting process recipes based on data from earlier processes, advanced model-based control methods, and self-updating process models.

Chapter 2 examined the optimal reactor design for an Area-Selective Atomic Layer Deposition reaction in a Discrete Feed Reactor and its impacts on conformal thin film deposition. A multiscale model was used to evaluate the efficacy of certain showerhead plates and determine optimal design criteria. It was found that a key step in conformal film growth is a focus on high initial concentration of reagent due to a statistically improbable precursor adsorption step. Thus, the combined-inlet reactor model completed the overall deposition process in the shortest process time.

Chapter 3 investigated the development of a process predictor model from aggregated process data. Four unique process datasets with varying kinetic parameters were generated with a first-principles multiscale model. Then, predictor models were trained on various combinations of these datasets and tested on unseen data with process noise to determine their efficacy. It was found that, for noisy datasets, the predictor model's performance improved when a broad range of process datasets were aggregated together. In contrast, when the test datasets were clean with little noise, the predictor model's performance was best when process datasets were not aggregated together.

Chapter 4 explored how a real-time Endpoint Feedback Controller and a classical Run-to-Run Controller interact when they are both trying to control the process time. The two systems were able to communicate through a β bias term. This bias term was applied to the Endpoint Controller and extended the process time by that amount after the controller determined that the process was complete. And in between each run, β is updated with an exponentially weighted moving average of a nonlinear model of the process through the Run-to-Run controller. The combination of both controllers outperformed either alone as the combination both minimized the immediate impact of process disturbances and efficiently drove the process to the target setpoint.

Chapter 5 investigated how to best create a data-driven process predictor for industrial process data collated from different tools in the same toolset. It also created a data aggregation method focusing on point-biserial correlations. It was found that tools with small datasets benefited greatly from data aggregation, with a total of three datasets being aggregated together yielding the best predictor model performance. Even for the largest datasets, aggregation still improved model performance, though to a lesser extent. Additionally, analyzing the various datasets by correlating them to each other with point-biserial correlations offered insights into the similarity of the two datasets. It was found that this correlation generally, but not always, let to data aggregation that improved model performance.

Bibliography

[1] 2022. Ansys Fluent User's Guide. ANSYS Inc., Canonsburg, PA.

[2] Ajayan, J., Nirmal, D., Tayal, S., Bhattacharya, S., Arivazhagan, L., Fletcher, A. A., Murugapandiyan, P., & Ajitha, D., 2021. Nanosheet field effect transistors-a next generation device to keep moore's law alive: An intensive study. Microelectronics Journal, 114, 105141.

[3] Appenzeller, J., Knoch, J., Bjork, M. T., Riel, H., Schmid, H., & Riess, W., 2008. Toward nanowire electronics. IEEE Transactions on Electron Devices, 55, 2827–2845.

[4] Bortz, A. B., Kalos, M. H., & Lebowitz, J. L., 1975. A new algorithm for Monte Carlo simulation of Ising spin systems. Journal of Computational Physics, 17, 10–18.

[5] Burg, D. & Ausubel, J. H., 2021. Moore's law revisited through Intel chip density. PLoS One, 16, e0256245.

[6] Chang, C. C., Pan, T. H., Wong, D. S. H., & Jang, S. S., 2011. A G&P EWMA algorithm for high-mix semiconductor manufacturing processes. Journal of Process Control, 21, 28–35.

[7] Cheimarios, N., To, D., Kokkoris, G., Memos, G., & Boudouvis, A. G., 2021. Monte Carlo and kinetic Monte Carlo models for deposition processes: A review of recent works. Frontiers in Physics, 9, 631918.

[8] Chen, T. C. T. & Chiu, M. C., 2022. Evaluating the sustainability of smart technology applications in healthcare after the COVID-19 pandemic: A hybridising subjective and objec-

tive fuzzy group decision-making approach with explainable artificial intelligence. Digital Health, 8, 20552076221136381.

[9] Christofides, P. D. & Armaou, A., 2006. Control and optimization of multiscale process systems. Computers & Chemical Engineering, 30, 1670–1686.

[10] Christofides, P. D., Davis, J. F., El-Farra, N. H., Clark, D., Harris, K. R., & Gipson, J. N., 2007. Smart plant operations: Vision, progress and challenges. AIChE Journal, 53, 2734–2741.

[11] Coleman, S., Kerr, D., & Zhang, Y., 2022. Image sensing and processing with convolutional neural networks. Sensors, 22, 3612.

[12] Correa, D., 2023. Global semiconductor production equipment market is expected to reach $209.9 billion by 2031. NASDAQ OMX's News Release Distribution Channel [Online], page 1.

[13] Crose, M., Zhang, W., Tran, A., & Christofides, P. D., 2018. Multiscale three-dimensional cfd modeling for PECVD of amorphous silicon thin films. Computers & Chemical Engineering, 113, 184–195.

[14] Del Castillo, E. & Hurwitz, A. M., 1997. Run-to-run process control: Literature review and extensions. Journal of Quality Technology, 29, 184–196.

[15] DeVita, J. P., Sander, L. M., & Smereka, P., 2005. Multiscale kinetic monte carlo algorithm for simulating epitaxial growth. Physical Review B, 72, 205421.

[16] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In North American Chapter of the Association for Computational Linguistics, 4171–4186, Minneapolis, MN, USA.

[17] Dobkin, D. M. & Zuraw, M. K. (Eds.), 2003. Principles of Chemical Vapor Deposition, Volume 1. Springer Dordrecht.

[18] Elers, K. E., Blomberg, T., Peussa, M., Aitchison, B., Haukka, S., & Marcus, S., 2006. Film uniformity in atomic layer deposition. Chemical Vapor Deposition, 12, 13–24.

[19] Elsayed, E. A., 2012. Overview of reliability testing. IEEE Transactions on Reliability, 61, 282–291.

[20] Fang, M. & Ho, J. C., 2015. Area-selective atomic layer deposition: Conformal coating, subnanometer thickness control, and smart positioning. ACS Nano, 9, 8651–8654.

[21] Fuchs, C., 2018. Industry 4.0: the digital german ideology. Triplec: Communication, Capitalism & Critique, 16, 280–289.

[22] George, S. M., 2010. Atomic layer deposition: An overview. Chemical Reviews, 110, 111–131.

[23] George, S. M., 2020. Mechanisms of thermal atomic layer etching. Accounts of Chemical Research, 53, 1151–1160.

[24] Gillespie, D. T., 1976. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. Journal of Computational Physics, 22, 403–434.

[25] Giménez, M. C., Del Pópolo, M. G., & Leiva, E. P. M., 2002. Kinetic Monte Carlo study of electrochemical growth in a heteroepitaxial system. Langmuir, 18, 9087–9094.

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

[27] Hong, S., An, N., Cho, H., Lim, J., Han, I. S., Moon, I., & Kim, J., 2023. A dynamic soft sensor based on hybrid neural networks to improve early off-spec detection. Engineering with Computers, 39, 3011–3021.

[28] Ingolfsson, A. & Sachs, E., 1993. Stability and sensitivity of an ewma controller. Journal of Quality Technology, 25, 271–287.

[29] Ishikawa, K., Karahashi, K., Ichiki, T., Chang, J. P., George, S. M., Kessels, W. M. M., Lee, H. J., Tinck, S., Um, J. H., & Kinoshita, K., 2017. Progress and prospects in nanoscale dry processes: How can we control atomic layer reactions? Japanese Journal of Applied Physics, 56, 06HA02.

[30] Islam, M., Chen, G., & Jin, S., 2019. An overview of neural network. American Journal of Neural Networks and Applications, 5, 7–11.

[31] Jansen, A. P. J. (Ed.), 2012. An Introduction to Kinetic Monte Carlo Simulations of Surface Reactions, Volume 1. Academic Press.

[32] Jiang, Y., Yin, S., Dong, J., & Kaynak, O., 2021. A review on soft sensors for monitoring, control, and optimization of industrial processes. IEEE Sensors Journal, 21, 12868–12881.

[33] Kadlec, P., Gabrys, B., & Strandt, S., 2009. Data-driven soft sensors in the process industry. Computers & Chemical Engineering, 33, 795–814.

[34] Kanarik, K. J., Lill, T., Hudson, E. A., Sriraman, S., Tan, S., Marks, J., Vahedi, V., & Gottscho, R. A., 2015. Overview of atomic layer etching in the semiconductor industry. Journal of Vacuum Science & Technology A, 33, 020802.

[35] Kanarik, K. J., Osowiecki, W. T., Lu, Y. J., Talukder, D., Roschewsky, N., Park, S. N., Kamon, M., Fried, D. M., & Gottscho, R. A., 2023. Human–machine collaboration for improving semiconductor process development. Nature, 616, 707–711.

[36] Kim, D. S., Kim, J. B., Ahn, D. W., Choe, J. H., Kim, J. S., Jung, E. S., & Pyo, S. G., 2023. Atomic layer etching applications in nano-semiconductor device fabrication. Electronics, Magnetics and Photonics, 19, 424–441.

[37] Kim, H. G., Kim, M., Gu, B., Khan, M. R., Ko, B. G., Yasmeen, S., Kim, C. S., Kwon, S. H., Kim, J., Kwon, J., Jin, K., Cho, B., Chun, J. S., Shong, B., & Lee, H. B. R., 2020. Effects of Al precursors on deposition selectivity of atomic layer deposition of Al_2O_3 using ethanethiol inhibitor. Chemistry of Materials, 32, 8921–8929.

[38] Kim, J., Chakrabarti, K., Lee, J., Oh, K. Y., & Lee, C., 2003. Effects of ozone as an oxygen source on the properties of the Al_2O_3 thin films prepared by atomic layer deposition. Materials Chemistry and Physics, 78, 733–738.

[39] King, R. D., Orhobor, O. I., & Taylor, C. C., 2021. Cross-validation is safe to use. Nature Machine Intelligence, 3, 276–276.

[40] Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization, 2017.

[41] Klement, P., Anders, D., Gümbel, L., Bastianello, M., Michel, F., Schörmann, J., Elm, M. T., Heiliger, C., & Chatterjee, S., 2021. Surface diffusion control enables tailored-aspect-ratio nanostructures in area-selective atomic layer deposition. ACS Applied Materials & Interfaces, 13, 19398–19405.

[42] Kondati Natarajan, S. & Elliott, S. D., 2018. Modeling the chemical mechanism of the thermal atomic layer etch of aluminum oxide: A density functional theory study of reactions during HF exposure. Chemistry of Materials, 30, 5912–5922.

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

[44] Li, Z. J. & Ren, T. L., 2012. Information Electronics in the Nanotechnology Era, 3–43. Springer Berlin Heidelberg, Berlin, Germany.

[45] Lin, S. C., Wang, C. C., Tien, C. L., Tung, F. C., Wang, H. F., & Lai, S. H., 2023. Fabrication

of aluminum oxide thin-film devices based on atomic layer deposition and pulsed discrete feed method. Micromachines, 14.

[46] Lin, T., Wang, Y., Liu, X., & Qiu, X., 2022. A survey of transformers. AI Open, 3, 111–132.

[47] Liu, Y., Zhang, Y., Wang, Y., Hou, F., Yuan, J., Tian, J., Zhang, Y., Shi, Z., Fan, J., & He, Z., 2024. A survey of visual transformers. IEEE Transactions on Neural Networks and Learning Systems, 35, 7478–7498.

[48] Mackus, A. J. M., Merkx, M. J. M., & Kessels, W. M. M., 2019. From the bottom-up: Toward area-selective atomic layer deposition with high selectivity. Chemistry of Materials, 31, 2–12.

[49] Mameli, A., Merkx, M. J. M., Karasulu, B., Roozeboom, F., Kessels, W. E. M. M., & Mackus, A. J. M., 2017. Area-selective atomic layer deposition of SiO_2 using acetylacetone as a chemoselective inhibitor in an ABC-type cycle. ACS Nano, 11, 9303–9311.

[50] Maroudas, D., 2000. Multiscale modeling of hard materials: Challenges and opportunities for chemical engineering. AIChE Journal, 46, 878–882.

[51] Mei, Z., Luo, Y., Qiao, Y., & Chen, Y., 2024. A novel joint segmentation approach for wafer surface defect classification based on blended network structure. Journal of Intelligent Manufacturing, 35, 1–15.

[52] Merkx, M. J. M., Sandoval, T. E., Hausmann, D. M., Kessels, W. M. M., & Mackus, A. J. M., 2020. Mechanism of precursor blocking by acetylacetone inhibitor molecules during area-selective atomic layer deposition of SiO_2. Chemistry of Materials, 32, 3335–3345.

[53] Moore, G. E., 1998. Cramming more components onto integrated circuits. Proceedings of the IEEE, 86, 82–85.

[54] Moyne, J., 2015. Run-to-Run Control in Semiconductor Manufacturing, 1248–1254. Springer London, London.

[55] Moyne, J., Qamsane, Y., Balta, E. C., Kovalenko, I., Faris, J., Barton, K., & Tilbury, D. M., 2020. A requirements driven digital twin framework: Specification and opportunities. IEEE Access, 8, 107781–107801.

[56] Mukesh, S. & Zhang, J., 2022. A review of the gate-all-around nanosheet fet process opportunities. Electronics, 11.

[57] Nguyen, P., Ivanov, D., & Sgarbossa, F., 2023. A digital twin–based approach to reinforce supply chain resilience: Simulation of semiconductor shortages. In Proceeding of IFIP International Conference on Advances in Production Management Systems, 563–576, Trondheim, Norway. Springer.

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

[59] Oh, I. K., Sandoval, T. E., Liu, T. L., Richey, N. E., & Bent, S. F., 2021. Role of precursor choice on area-selective atomic layer deposition. Chemistry of Materials, 33, 3926–3935.

[60] Orji, N. G., Badaroglu, M., Barnes, B. M., Beitia, C., Bunday, B. D., Celano, U., Kline, R. J., M. Neisser, Y. O., & Vladar, A. E., 2018. Metrology for the next generation of semiconductor devices. Nature Electronics, 1, 532–547.

[61] Ou, F., Abdullah, F., Wang, H., Tom, M., Orkoulas, G., & Christofides, P. D., 2024. Sparse identification modeling and predictive control of wafer temperature in an atomic layer etching reactor. Chemical Engineering Research and Design, 202, 1–11.

[62] O'Donovan, P., Leahy, K., Bruton, K., & O'Sullivan, D. T., 2015. Big data in manufacturing: a systematic mapping study. Journal of Big Data, 2, 1–22.

[63] Pan, D., Li, T., Chien Jen, T., & Yuan, C., 2014. Numerical modeling of carrier gas flow in atomic layer deposition vacuum reactor: A comparative study of lattice boltzmann models. Journal of Vacuum Science & Technology A, 32, 01A110.

[64] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

[65] Perera, Y. S., Ratnaweera, D., Dasanayaka, C. H., & Abeykoon, C., 2023. The role of artificial intelligence-driven soft sensors in advanced sustainable process industries: A critical review. Engineering Applications of Artificial Intelligence, 121, 105988.

[66] Platzer, M. D. & Sargent, J. F., 2016. US semiconductor manufacturing: Industry trends, global competition, Federal Policy. Congressional Research Service New York.

[67] Radamson, H., 2018. 1 - Basics of metal–oxide–semiconductor field-effect transistor (MOSFET). In Radamson, H. H., Luo, J., Simoen, E., & Zhao, C. (Eds.), CMOS Past, Present and Future, 1–17. Woodhead Publishing.

[68] Ranganathan, G., 2021. A study to find facts behind preprocessing on deep learning algorithms. Journal of Innovative Image Processing, 3, 66–74.

[69] Rehmer, A. & Kroll, A., 2020. On the vanishing and exploding gradient problem in gated recurrent units. IFAC-PapersOnLine, 53, 1243–1248.

[70] Ren, Y. M., Alhajeri, M. S., Luo, J., Chen, S., Abdullah, F., Wu, Z., & Christofides, P. D.,

2022. A tutorial review of neural network modeling approaches for model predictive control. Computers & Chemical Engineering, 165, 107956.

[71] Roland, J. P., Marcoux, P. J., Ray, G. W., & Rankin, G. H., 05 1985. Endpoint detection in plasma etching. Journal of Vacuum Science & Technology A, 3, 631–636.

[72] Sarker, I. H., 2021. Machine learning: Algorithms, real-world applications and research directions. SN Computer Science, 2, 160.

[73] Seo, S., Yeo, B. C., Han, S. S., Yoon, C. M., Yang, J. Y., Yoon, J., Yoo, C., Kim, jin H., Lee, baek Y., Lee, S. J., Myoung, J. M., Lee, H. B. R., Kim, W. H., Oh, I. K., & Kim, H., 2017. Reaction mechanism of area-selective atomic layer deposition for Al_2O_3 nanopatterns. ACS Applied Materials & Interfaces, 9, 41607–41617.

[74] Shao, G., Jain, S., Laroque, C., Lee, L. H., Lendermann, P., & Rose, O., 2019. Digital twin for smart manufacturing: The simulation aspect. In 2019 Winter Simulation Conference (WSC), 2085–2098, National Harbor, MD, USA.

[75] Shauly, E. N., 2023. Design rules in a semiconductor foundry / edited by Eitan N. Shauly. Jenny Stanford Publishing, Singapore.

[76] Singh, M., Sargent, J. F., & Sutter, K. M. Semiconductors and the semiconductor industry / Manpreet Singh, John F. Sargent Jr., Karen M. Sutter., 2023.

[77] Songkhla, S. N. & Nakamoto, T., 2021. Overview of quartz crystal microbalance behavior analysis and measurement. Chemosensors, 9, 350.

[78] Steinkraus, D., Buck, I., & Simard, P., 2005. Using gpus for machine learning algorithms. In Eighth International Conference on Document Analysis and Recognition (ICDAR'05), Seoul, South Korea.

[79] Sun, C., Rose, T., Ehm, H., & Heilmayer, S., 2015. Complexity management in the semiconductor supply chain and manufacturing using PROS analysis. In Liu, K., Nakata, K., Li, W., & Galarreta, D. (Eds.), Information and Knowledge Management in Complex Systems, 166–175, Cham, Switzerland. Springer International Publishing.

[80] Sun, H., Patel, V., Singh, B., Ng, C., & Whittaker, E., 1994. Sensitive plasma etching endpoint detection using tunable diode laser absorption spectroscopy. Applied Physics Letters, 64, 2779–2781.

[81] Sun, Q. & Ge, Z., 2021. A survey on deep learning for data-driven soft sensors. IEEE Transactions on Industrial Informatics, 17, 5853–5866.

[82] Tom, M., Yun, S., Wang, H., Ou, F., Orkoulas, G., & Christofides, P. D., 2022. Machine learning-based run-to-run control of a spatial thermal atomic layer etching reactor. Computers & Chemical Engineering, 168, 108044.

[83] Tom, M., Wang, H., Ou, F., Yun, S., Orkoulas, G., & Christofides, P. D., 2023. Computational fluid dynamics modeling of a discrete feed atomic layer deposition reactor: Application to reactor design and operation. Computers & Chemical Engineering, 178, 108400.

[84] Tom, M., Wang, H., Ou, F., Orkoulas, G., & Christofides, P. D., 2024. Machine learning modeling and run-to-run control of an area-selective atomic layer deposition spatial reactor. Coatings, 14, 1–7.

[85] Tseng, S. H., 2022. CMOS MEMS design and fabrication platform. Frontiers in Mechanical Engineering, 8, 894484.

[86] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I., 2017. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S.,

Wallach, H., Fergus, R., Vishwanathan, S., & Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Volume 30, 1–11, Long Beach, CA, USA. Curran Associates, Inc.

[87] Voas, J., Kshetri, N., & DeFranco, J. F., 2021. Scarcity and global insecurity: The semiconductor shortage. IT Professional, 23, 78–82.

[88] Wan, J., McLoone, S., English, P., O'Hara, P., & Johnston, A., 2014. Predictive maintenance for improved sustainability — an ion beam etch endpoint detection system use case. In Intelligent Computing in Smart Grid and Electrical Vehicles, 147–156, Berlin, Heidelberg.

[89] Wang, C. P., Tsai, Y. P., Lin, B. J., Liang, Z. Y., Chiu, P. W., Shih, J. R., Lin, C. J., & King, Y. C., 2020. On-wafer finfet-based EUV/eBeam detector arrays for advanced lithography processes. IEEE Transactions on Electron Devices, 67, 2406–2413.

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

[91] Wang, H., Ou, F., Suherman, J., Tom, M., Orkoulas, G., & Christofides, P. D., 2024. Data-driven machine learning predictor model for optimal operation of a thermal atomic layer etching reactor. Industrial & Engineering Chemistry Research, 63, 19693–19706.

[92] Wang, H., Tom, M., Ou, F., Orkoulas, G., & Christofides, P. D., 2024. Multiscale computational fluid dynamics modeling of an area-selective atomic layer deposition process using a discrete feed method. Digital Chemical Engineering, 10, 100140.

[93] Waoo, A. A. & Soni, B. K., 2021. Performance analysis of sigmoid and relu activation functions in deep neural network. In Intelligent Systems: Proceedings of SCIS 2021, 39–52, Jaipur, India. Springer.

[94] Wehinger, G. D., Ambrosetti, M., Cheula, R., Ding, Z. B., Isoz, M., Kreitz, B., Kuhlmann, K., Kutscherauer, M., Niyogi, K., Poissonnier, J., Réocreux, R., Rudolf, D., Wagner, J., Zimmermann, R., Bracconi, M., Freund, H., Krewer, U., & Maestri, M., 2022. Quo vadis multiscale modeling in reaction engineering? – A perspective. Chemical Engineering Research and Design, 184, 39–58.

[95] Wu, B. & Kumar, A., 10 2007. Extreme ultraviolet lithography: A review. Journal of Vacuum Science & Technology B: Microelectronics and Nanometer Structures Processing, Measurement, and Phenomena, 25, 1743–1761.

[96] Wu, Z., Tran, A., Rincon, D., & Christofides, P. D., 2019. Machine learning-based predictive control of nonlinear processes. part i: Theory. AIChE J., 65, e16729.

[97] Xie, W. & Parsons, G. N., 2020. Thermal atomic layer etching of metallic tungsten via oxidation and etch reaction mechanism using O_2 or O_3 for oxidation and WCl_6 as the chlorinating etchant. Journal of Vacuum Science & Technology A, 38, 022605.

[98] Xu, B., Wang, N., Chen, T., & Li, M., 2015. Empirical evaluation of rectified activations in convolutional network. arXiv preprint arXiv:1505.00853.

[99] Yuan, C. Y. & Sheng, Y., 2010. Sustainable scale-up studies of atomic layer deposition for microelectronics manufacturing. In Proceedings of the 2010 IEEE International Symposium on Sustainable Systems and Technology, 1–6, Arlington, VA, USA.

[100] Yun, S., Tom, M., Luo, J., Orkoulas, G., & Christofides, P., 2021. Microscopic and data-driven modeling and operation of thermal atomic layer etching of aluminum oxide thin films. Chemical Engineering Research and Design, 177, 96–107.

[101] Yun, S., Ou, F., Wang, H., Tom, M., Orkoulas, G., & Christofides, P. D., 2022. Atomistic-mesoscopic modeling of area-selective thermal atomic layer deposition. Chemical Engineering Research & Design, 188, 271–286.

[102] Yun, S., Tom, M., Luo, J., Orkoulas, G., & Christofides, P. D., 2022. Microscopic and data-driven modeling and operation of thermal atomic layer etching of aluminum oxide thin films. Chemical Engineering Research & Design, 177, 96–107.

[103] Yun, S., Tom, M., Ou, F., Orkoulas, G., & Christofides, P. D., 2022. Multiscale computational fluid dynamics modeling of thermal atomic layer etching: Application to chamber configuration design. Computers & Chemical Engineering, 161, 107757.

[104] Yun, S., Tom, M., Ou, F., Orkoulas, G., & Christofides, P. D., 2022. Multivariable run-to-run control of thermal atomic layer etching of aluminum oxide thin films. Chemical Engineering Research and Design, 182, 1–12.

[105] Yun, S., Wang, H., Tom, M., Ou, F., Orkoulas, G., & Christofides, P. D., 2023. Multiscale CFD modeling of area-selective atomic layer deposition: Application to reactor design and operating condition calculation. Coatings, 13, 558.

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

[107] Zhang, J., Li, Y., Cao, K., & Chen, R., 2022. Advances in atomic layer deposition. Nanomanufacturing and Metrology, 5, 191–208.

[108] Zhang, L., Wang, S., Ma, C., He, J., Xu, C., Ma, Y., Ye, Y., Liang, H., Chen, Q., & Chan, M., 2012. Gate underlap design for short channel effects control in cylindrical gate-all-around MOSFETs based on an analytical model. IETE Technical Review, 29, 29–35.

[109] Zhang, Y., Ding, Y., & Christofides, P. D., 2020. Multiscale computational fluid dynamics modeling and reactor design of plasma-enhanced atomic layer deposition. Computers & Chemical Engineering, 142, 107066.

[110] Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., & others, , 2023. A survey of large language models. arXiv preprint arXiv:2303.18223.

[111] Zheng, P., Connelly, D., Ding, F., & Liu, T. J. K., 2015. FinFET evolution toward stacked-nanowire FET for CMOS technology scaling. IEEE Transactions on Electron Devices, 62, 3945–3950.