A DATA DRIVEN DIGITAL TWIN APPROACH TO OPTIMIZE CONTINUOUS PRODUCTION ENVIRONMENT WITH DEEP REINFORCEMENT LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

ALİ HAYDAR SİVRİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

SEPTEMBER 2023

A DATA DRIVEN DIGITAL TWIN APPROACH TO OPTIMIZE CONTINUOUS PRODUCTION ENVIRONMENT WITH DEEP REINFORCEMENT LEARNING

Submitted by Ali Haydar Sivri in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems Department, Middle East Technical University** by,

Date: *06.09.2023*

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Ali Haydar Sivri

Signature : _____

ABSTRACT

A DATA DRIVEN DIGITAL TWIN APPROACH TO OPTIMIZE CONTINUOUS PRODUCTION ENVIRONMENT WITH DEEP REINFORCEMENT LEARNING

Sivri, Ali Haydar
MSc., Department of Information Systems
Supervisor: Prof. Dr. Altan Koçyiğit
Co-supervisor: Dr. Mert Onuralp Gökalp

September 2023, 67 pages

Today, the world mainly strives to minimize carbon emissions and maximize efficiency in energy production to lower energy costs and greenhouse effect. Therefore, optimizing hydrogen production processes becomes a crucial objective in organizations, especially for manufacturers that consume significant energy in production. Organizations should operate their production processes under the most optimum conditions to ensure their survival and attain competitive advantage. However, they face substantial challenges in optimizing their energy utilizations in their production processes. Those challenges consequently necessitate utilizing a data-driven approach and dynamic framework for managing and optimizing their process workflows. This thesis study consists of two main phases. The first phase focuses on developing a data-driven simulation environment (DDSE) of the hydrogen unit for Deep Reinforcement Learning (DRL) that can adapt to the complexity and variability of processes by utilizing the machine learning approaches. The DDSE comprises the intricacies of the hydrogen production process, including critical parameters such as temperature, pressure, and catalyst variables. At the second phase, advanced DRL algorithms are developed to support continuous learning and making decisions based on long-term outcomes rather than immediate rewards. The DRL model effectively interacts with the DDSE in adjusting controllable production parameters to optimize the hydrogen production yield. The proposed model's validation and efficacy are demonstrated through extensive simulations and real-world data. The test results reveal that we can improve energy efficiency by approximately 4%.

Keywords: Machine Learning, Deep Reinforcement Learning, Data Driven Simulation, Hydrogen Production, Optimization

ÖZ

DERİN PEKİŞTİRMELİ ÖĞRENME İLE SÜREKLİ ÜRETİM ORTAMINI OPTİMİZE ETMEYE YÖNELİK VERİ ODAKLI DİJİTAL İKİZ YAKLAŞIMI

Sivri, Ali Haydar

Yüksek Lisans, Bilişsel Bilimler Bölümü

Tez Yöneticisi: Prof. Dr. Altan Koçyiğit

Ortak Tez Yöneticisi: Dr. Mert Onuralp Gökalp

Eylül 2023, 67 sayfa

Bugün dünya, enerji maliyetlerini ve sera etkisini azaltmak için enerji üretiminde karbon emisyonlarını minimize etmeyi ve verimliliği maksimize etmeyi hedeflemektedir. Bu bağlamda, üretimde önemli enerji tüketen enerji üreticileri için hidrojen üretim süreçlerini optimize etmek kritik bir hedef haline gelir. Kuruluşlar, rekabet avantajı elde edebilmek ve hayatta kalabilmek için üretim süreçlerini en optimum koşullar altında işletmelidir. Ancak, üretim süreçlerinde enerji kullanımlarını optimize etmede önemli zorluklarla karşılaşırlar. Bu zorluklar, süreç akışlarını yönetmek ve optimize etmek için veriye dayalı bir yaklaşım ve dinamik bir çerçeve kullanmayı zorunlu kılar. Bu tez çalışması iki ana aşamadan oluşmaktadır. İlk aşama, ML yaklaşımlarını kullanarak süreçlerin karmaşıklığına ve değişkenliğine uyum sağlayabilen Derin Pekiştirmeli Öğrenme (DRL) için bir veriye dayalı simülasyon ortamı (DDSE) geliştirmeye odaklanır. DDSE, hidrojen üretim sürecinin karmaşıklıklarını, sıcaklık, basınç ve katalizör değişkenleri gibi kritik parametreleri içerir. İkinci aşamada, sürekli öğrenmeyi desteklemek ve uzun vadeli sonuçlara dayalı kararlar almak için gelişmiş DRL algoritmaları geliştirilir. DRL modeli, hidrojen üretim verimini optimize etmek için kontrol edilebilir üretim parametrelerini ayarlamada DDSE ile etkili bir şekilde etkileşim kurar. Önerilen modelin doğrulama ve etkinliği, kapsamlı simülasyonlar ve gerçek dünya verileri aracılığıyla gösterilmiştir. Test sonuçları, enerji verimliliğini yaklaşık olarak %4 artırabileceğimizi göstermektedir.

Anahtar Sözcükler: Makine Öğrenmesi, Derin Pekiştirmeli Öğrenme, Veri Odaklı Simülasyon, Hidrojen Üretimi, Optimizasyon

To My Family

ACKNOWLEDGMENTS

First and foremost, I would like to thank my supervisor Prof. Dr. Altan Koçyiğit for his suggestions, support, encouragement, and for sharing his experiences and knowledge through this research. I am very grateful for his inspiring ideas. He always provides insightful and enlightening discussions about research. He has given me warm support, valuable guidance, helpful advice, time, and shown great patience.

I would like to express my sincere appreciation to my co-supervisor, Dr. Mert Onuralp Gökalp. He has given me warm support, priceless guidance, generous advice, continuous support, and shown great patience through emails and phone calls than I ever would expect from a co-advisor.

I would like to thank my warm-hearted, dedicated, and helpful wife, Talya Tümer Sivri, for her emotional and physical support with endless tolerance and love while sharing the home and life with me throughout my thesis process. Thank you for loving me endlessly and believing in me.

Finally, I would like to express my very special gratitude to my parents, my father, Kemal Sivri, my mother, Serican Sivri, and my sister, Serap Sivri, for their exceptional patience, love, encouragement, and endless support.

TABLE OF CONTENTS

ABSTRACT iv
ÖZ v
DEDICATION vi
ACKNOWLEDGMENTS vii
TABLE OF CONTENTS viii
LIST OF TABLES x
LIST OF FIGURES xi
LIST OF ABBREVIATIONS xii
CHAPTERS
1. INTRODUCTION 1
2. BACKGROUND AND RELATED WORK 5
2.1. Importance of Hydrogen in the Energy Landscape 5
2.2. Hydrogen Production in Oil and Gas Sector 5
2.3. Data Driven Digital Twin 6
2.4. Machine Learning and Deep Reinforcement Learning 6
2.4.1. Introduction to Machine Learning 6
2.4.2. Linear Regression 7
2.4.3. XGBoost 7
2.4.4. LightGBM 7
2.4.5. Feed-Forward Neural Networks 7
2.4.6. LASSO and Ridge Regression 8
2.4.7. Deep Reinforcement Learning 8
2.4.8. Key Components of Reinforcement Learning 9
2.4.9. Task Type in Reinforcement Learning 10
2.4.10. The Exploration/Exploitation Trade-off 10
2.4.11. Proximal Policy Optimization (PPO) 10

2.4.12. Soft-Actor-Critic (SAC) 11
2.4.13. Deep Deterministic Policy Gradient (DDPG) 12
2.5. Related Work 13
3. METHODOLOGY and RESULTS 17
3.1. Data Sources 18
3.2. Data Preprocessing 21
3.2.1. Outlier and Shutdown Elimination 22
3.2.2. Missing Value 22
3.2.3. Scaling 22
3.2.4. Feature Extraction 23
3.2.5. Feature Selection 23
3.3. Multiple Model Training 25
3.3.1. Modelling Part 26
3.4. Deep Reinforcement Learning Models 40
3.4.1. Reward 42
3.5. Results 43
4. CONCLUSION 47
REFERENCES 49
APPENDICES 55
APPENDIX-A RESULTS OF FIVE FOLD CROSS VALIDATION WITHOUT RANDOM SPLITTING 55
APPENDIX-B SELECTED FEATURES FOR EACH TARGET AFTER THE FEATURE SELECTION 59

LIST OF TABLES

Table 1. Comparison table between PPO, SAC and DDPG 13
Table 2. Feature names and their sampling rates. 19
Table 3. Base models default hyperparameters.....27
Table 4. Models' train and test results in terms of RMSE and RMSE/STD 29
Table 5. Model results with feature engineering.....32
Table 6. The optimal hyperparameter setups for 16 different targets (All of them are XGBoost models).....38
Table 7. DRL models' parameters.....40
Table 8. Result table considering the time axis.....55
Table 9. Final selected features for each targets. 59

LIST OF FIGURES

Figure 1. The RL process: a loop of state, action, reward and next state [22]. 9
Figure 2. General framework 18
Figure 3. Data-driven simulation structure flow. 21
Figure 4. The selected number of features for each target. 25
Figure 5. PPO, SAC, DDPG episode rewards for 10000. 44
Figure 6. PPO, SAC, DDPG episode rewards for 100000. 45
Figure 7. When the SAC agent is in the environment to optimize pressure value. 45

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
RL	Reinforcement Learning
DRL	Deep Reinforcement Learning
DQL	Deep Q-Learning
ML	Machine Learning
DDS	Data-Driven Simulation
DDSE	Data-Driven Simulation Environment
IoT	Internet of Things
IIoT	Industrial Internet of Things
GDP	Gross Domestic Product
RQ	Research Question
XGBoost	Extreme Gradient Boosting
LightGBM	Light Gradient Boosting Machine
FFNN	Feed-Forward Neural Network
ANN	Artificial Neural Network
LASSO	Least Absolute Shrinkage and Selection Operator
SMR	Steam Methane Reforming
CH4	Methane
H2O	Steam
CO	Carbon monoxide
CO2	Cardon dioxide
H	Hydrogen
GOSS	Gradient-based One-Side Sampling
EFB	Exclusive Feature Bundling
MDP	Markov Decision Process
PPO	Proximal Policy Optimization
SAC	Soft Actor-Critic
PCB	Printed Circuit Boards
DDPG	Deep Deterministic Policy Algorithm

CHAPTER 1

INTRODUCTION

Since the industrial revolution, manufacturing has been a key contributor to worldwide economic expansion. According to the World Bank data the manufacturing sector accounted for 16% of the global GDP in 2018, offering employment to over 400 million individuals [1]. Furthermore, manufacturing activities claim responsibility for 22% of the world's energy utilization and 40% of total carbon emissions [2]. Due to these implications, there is an increasing interest in integrating novel technologies to augment manufacturing operations' efficiency and shorten associated expenses. The dilemma associated with energy is increasingly intensifying, marked by the continuous augmentation of production and consumption expenditures. Given these circumstances, manufacturing entities diligently optimize their production processes to maintain efficiency under these challenging conditions. The recent research investigates that 70% of top-level executives affirmed that engaging in data-centric analytics is vital for their organizations to attain a competitive edge [3]. Consequently, in the era of big data, especially manufacturing organizations need to prioritize a data-driven approach for their processes to secure their competitive position in the market, utilizing information technologies and Internet of Things (IoT) devices [4]. The advancements in digital transformation, data analytics and predictive modeling domain enables widespread adoption of the novel technologies that increases computational capabilities, and the utilization of the Industrial Internet of Things (IIoT) [5]. In recent years, a remarkable wave in digital transformation across diverse sectors has been observed, characterized by the progressive implementation of digital technologies to augment efficiency and stimulate innovation. This paradigmatic transformation has been amplified as companies accumulate a wealth of data from multiple origins, encompassing sensors. Numerous case studies have evidenced that enhanced oversight of manufacturing procedures can yield significant fiscal savings, frequently amounting to millions [6].

The expansion of recent technologies including IIoT, big data, and digital transformation has made data a fundamental impact on organizations. Today, manufacturing organizations especially can gather, store and process a higher volume of data from various sources than ever before. These advancements allow both researchers and practitioners to design and realize the smart manufacturing concept in a fully manner [7]. Accordingly, effectively utilizing the big data generated from IIoT devices and historical big process data, we can build analytics models that represent a selected period's normality and compare any variations in real-time [8]. Moreover, by effectively leveraging big data

analytics throughout the organization, we can derive actionable insights from such data to improve product quality, reduce waste, and increase profits. Data analytics presents significant opportunities for these organizations to gain a competitive edge in their marketplace, and also generating innovative revenue channels by enabling them to transform raw data into manufacturing intelligence.

In manufacturing, executing processes with maximum efficiency plays a significant role in maintaining competitive integrity within the market landscape. Specifically, the escalating energy and raw materials consumption is progressively becoming a substantial challenge, consequently inflating the associated production expenses. Nevertheless, enterprises encounter difficulties when attempting to optimize their complex procedures. In order to manage these intricacies, they resort to simplifying assumptions that inadvertently cause them to overlook certain nuanced aspects within their operations, leading to missed opportunities for optimal solutions.

Especially oil and gas refineries, the process becomes even more complicated, and finding the optimal operating conditions becomes much more difficult. Most contemporary industrial optimization and control systems rely heavily on models, necessitating superior quality process models for effective performance. It is conventional to perpetually scrutinize system performance and instigate a corrective model re-identification process in case of any performance downturns. The re-identification phase can span over two weeks or more, often necessitating the application of external stimuli, which imposes a costly disruption to the regular workflow of the process. This re-identification is particularly complex for multi-variable processes, requiring a unique model for every combination of inputs and outputs [9], [10].

Therefore, there is a need for a framework that captures data-driven, dynamic, and complex non-linear relationships to model and optimize the process. In addition, the models used in the process optimization require a lot of domain knowledge and they are challenging to design, develop and manage. However, the data-driven modelling and model-free deep RL approaches can provide significant contribution to optimize these processes by minimizing domain knowledge gap. Accordingly, this thesis seeks to answer two important research questions as follows:

RQ – 1: Could I construct a data-driven simulation model of the hydrogen production unit using state-of-the-art machine learning approaches?

RQ – 2: Could I optimize hydrogen production using model-free Deep Reinforcement Learning based on the results of the data-driven simulation we developed?

This thesis focuses on developing and implementing a unique framework to optimize hydrogen production in oil refineries. The central objective is to enhance production yield and efficiency by intelligently navigating the complex, high-dimensional space of possible process configurations. The Machine Learning (ML) model was developed to simulate hydrogen production unit in a data-driven manner. The DRL model is designed to interact

with an environment that accurately models the intricacies of the hydrogen production process, including critical variables such as temperature, pressure, and catalyst usage. The scope of this thesis comprises two main phases (1) developing a simulation environment of the hydrogen production unit in a data-driven manner and, (2) developing a Deep Reinforcement Learning (DRL) model by utilizing Data-Driven Simulation (DDS) model developed in the first phase as an environment and evaluating its performance in simulation scenarios and real-world data.

This thesis consists of four chapters. After this introductory chapter, Chapter 2 aims to give an overview of Hydrogen production, ML, and DRL. Chapter 3 describes the methodology and results. In Chapter 4 the study is summarized, and the conclusion is described.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, I give background information in the scope of this thesis study. Section 2.1. provides an overview of hydrogen production. I explain the ML and DRL concepts in Section 2.2 and 2.3 respectively. Finally, I review existing studies related to data driven process optimization in Section 2.4.

2.1. Importance of Hydrogen in the Energy Landscape

Hydrogen, often dubbed as the energy source of tomorrow, has been accumulating a notable amount of interest in recent times due to its broad-based uses and its capacity to usher in a greener energy horizon. Being a transparent, scent-free, and high-performance fuel, hydrogen can be employed in fuel cells to produce electricity, which in turn curtails dependence on fossil fuels [11]. In the realm of industrial utilities, hydrogen occupies an indispensable position, notably in petroleum refineries where it is instrumental in manufacturing lighter and purer fuels by eliminating sulfur content from raw oil [12].

2.2. Hydrogen Production in Oil and Gas Sector

Hydrogen is an integral element in the oil and gas sector, primarily utilized in refineries for various procedures, including hydrotreating and hydrocracking. These processes aim to lower sulfur levels, enhance fuel quality, and comply with environmental standards. As refineries grapple with the necessity to process denser and more sulfur-laden crude oil, alongside the growing demand for purer products, the requirement for hydrogen production in refineries is persistently escalating.

The most common method employed for hydrogen production in refineries is Steam Methane Reforming (SMR). This process entails the reaction of a hydrocarbon, typically methane (CH4), with steam (H2O) under considerable temperature (700-1100°C) and pressure conditions, resulting in the production of hydrogen and carbon monoxide [13]. This reaction is represented by:

$$CH_4 + H_2O \leftrightarrow \mathrm{CO} + 3H_2 \qquad (1)$$

This reaction is endothermic, implying that it necessitates heat.

The production of hydrogen in refineries is a multifaceted procedure fraught with various challenges. The energy consumption of the Steam Methane Reforming (SMR) process is consequential due to the high temperature required. Additionally, the process heads to the production of a noteworthy amount of CO, contributing to the emission of greenhouse gases.

Optimization of this hydrogen production procedure involves enhancing hydrogen yield and minimizing energy usage and CO emissions. These complexities are further compounded by the process's dynamic character, encompassing many variables and non-linear reactions.

2.3. Data Driven Digital Twin

A data-driven digital twin represents a convergence of simulation and real-world data to create a virtual representation of a physical asset or process [14]. By continuously updating the digital twin with real-time data, it dynamically reflects the current and historical state of its physical counterpart. Unlike traditional simulation models, which are largely based on theoretical assumptions or idealized scenarios, a data-driven digital twin evolves based on actual performance data [15]. This not only enhances predictive accuracy but also enables rapid response to changes, making it invaluable for monitoring, diagnostics, optimization, and predictive maintenance in industries ranging from manufacturing to healthcare. The integration of data-driven insights with a digital twin's framework paves the way for more informed decision-making, leading to improved operational efficiency and innovation [16].

2.4. Machine Learning and Deep Reinforcement Learning

This chapter aims to discuss various ML methodologies, specifically focusing on linear regression, XGBoost (Extreme Gradient Boosting), LightGBM (Light Gradient Boosting Machine), Feed-Forward Neural Networks (FFNN), and regularization techniques like Lasso and Ridge regression. These methods are integral in predictive analytics, holding considerable potential for a wide range of applications.

2.4.1. Introduction to Machine Learning

ML, a crucial component of Artificial Intelligence (AI), centers on creating systems capable of data-driven learning and decision-making [17]. ML methodologies gather knowledge from exposure and progressively refine their performance as they process more data over time. These methodologies can be primarily categorized into three types: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning is a process that involves understanding a function that links an input to its corresponding output, drawing on sets of example pairs. This approach has seen wide application in fields such as regression and classification [18]. Conversely, unsupervised learning identifies patterns in input data without requiring clearly defined output labels. This technique is frequently employed for clustering and dimensionality reduction tasks [19].

2.4.2. Linear Regression

Linear regression, one of the most established statistical techniques, establishes a relationship between a dependent variable and one or more independent variables [20]. The essence of linear regression lies in its ability to infer the conditional expectation of the dependent variable given the independent variables. Although simple, linear regression often serves as a fundamental component of more complex models [21].

2.4.3. XGBoost

XGBoost is an implementation of gradient boosting machines that has gained popularity for its efficiency and effectiveness. Gradient boosting constructs new predictors that aim to correct the residuals errors of the prior predictor, thus reducing the overall prediction error [22]. XGBoost's strength lies in its scalability in all scenarios and its ability to produce high-quality predictions.

2.4.4. LightGBM

LightGBM, developed by Microsoft, is a gradient boosting framework that uses tree-based learning algorithms. It is designed to be fast and efficient, with a focus on performance and speed. It introduces two novel techniques: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) [23]. These enable LightGBM to handle larger datasets with higher efficiency compared to other gradient boosting models.

2.4.5. Feed-Forward Neural Networks

Feed-forward neural networks are a type of artificial neural network where the connections between the nodes do not form a cycle. This model's non-linearity enables it to capture complex patterns and relationships in data [19]. FFNNs have shown promising results in a variety of applications, particularly in areas involving unstructured data, such as image and language processing.

2.4.6. LASSO and Ridge Regression

Ridge and Lasso are two popular regularization techniques that prevent overfitting in regression models by adding a penalty term to the loss function. Ridge regression adds an L2 penalty term, which minimizes the coefficient values but does not reduce them to zero [24]. Conversely, Least Absolute Shrinkage and Selection Operator (LASSO) includes an L1 penalty term, which can shrink some coefficients to zero, effectively performing variable selection [25].

2.4.7. Deep Reinforcement Learning

The central aspiration of ML algorithms lies in shaping a system that can effectively navigate an uncertain environment. Among various ML methodologies, RL is acknowledged as one of the most proficient techniques for managing control problems [26]. In RL, an agent performs actions sequentially, each action leading to a transition from a current state to the next. The reward function, a key component, evaluates this transition and guides the agent towards long-term objectives, consequently helping to identify optimal actions. DRL combines RL and deep learning to enable an agent to learn and adapt its behavior based on its interactions with the environment.

Since the seminal paper on Deep Q-Learning [27], there have been substantial advancements in the field of DRL. Noteworthy examples include the achievement of OpenAI Five in competing against top-tier Dota2 players, and the Dexterity project, which signify the exciting progress in DRL research. RL has emerged as a powerful tool to solve control tasks, also known as decision problems, by developing agents that learn from trial-and-error interactions with the environment, with rewards serving as unique feedback.

RL, a prominent area in AI research, was developed in the domains of computer science and operations research to address complex sequential decision-making challenges [26]. The goal in RL is to identify an optimal policy through the interaction of an agent (such as a controller) with a stochastic environment (such as a plant), considered as a Markov decision process (MDP). The recent resurgence of interest in RL is attributed to the successful integration of RL with deep learning, allowing effective generalization to MDPs with continuous state spaces [28], see Figure 1.

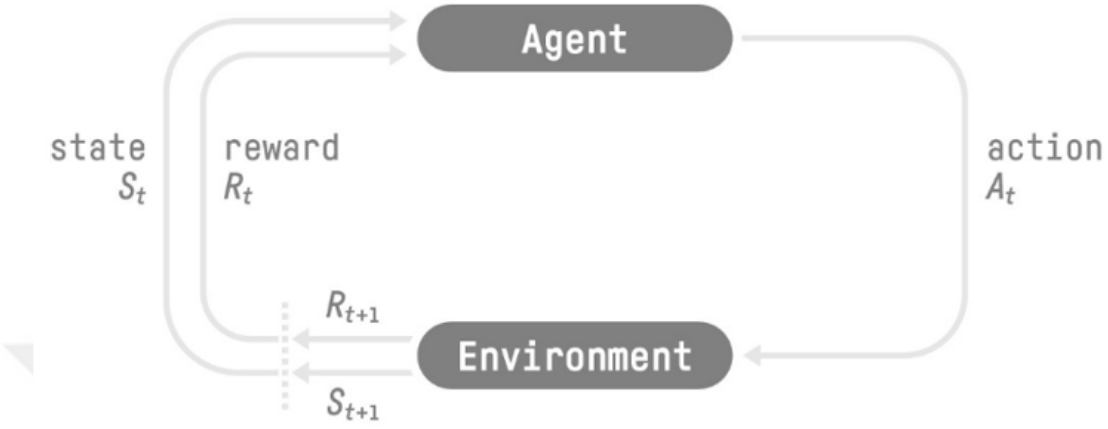

Figure 1. The RL process: a loop of state, action, reward and next state [22].

2.4.8. Key Components of Reinforcement Learning

The RL framework involves an interaction loop of state, action, reward, and the next state. Each component can be defined as follows [29]:

• **Reward:** A scalar value received sporadically from the environment, serving as feedback on the agent's behavior.

• **Agent:** An entity that interacts with the environment through specific actions, observations, and rewards.

• **Environment:** The context within which the agent operates, defined by the rewards it offers, the actions it permits, and the observations it affords.

• **Action:** The agent's interactions with the environment, which could be discrete or continuous.

• **Observation:** Information about the immediate surroundings, obtained by the agent from the environment.

The agent's objective is to maximize its cumulative reward or expected return.

2.4.9. Task Type in Reinforcement Learning

In RL, tasks can be categorized into episodic and continuous tasks. Episodic tasks have defined start and end points, with the sequence of states, actions, and rewards forming an episode, for instance, a level in Super Mario Bros. On the other hand, continuous tasks persist indefinitely, with the agent needing to interact with the environment while selecting optimal actions, for example, an agent conducting automated stock trading [30].

2.4.10. The Exploration/Exploitation Trade-off

The exploration/exploitation trade-off is an essential consideration in RL. Exploration involves the agent conducting random actions to understand the environment better, while exploitation utilizes known information to maximize rewards.

2.4.11. Proximal Policy Optimization (PPO)

Proximal Policy Optimization is a type of policy optimization method which is part of the larger family of Policy Gradient methods for RL [31]. Unlike previous policy gradient methods, PPO aims to take the largest possible improvement step, while ensuring the new policy is not too far from the old policy. This characteristic prevents the detrimental effects of large updates, enabling more stable and efficient learning. The PPO method is robust and relatively easy to implement, which has made it a standard algorithm in DRL.

The algorithm's objective is to uncover a policy that optimizes the accumulated reward over an extended time span. PPO functions by cyclically gathering data from the environment and modifying the policy in accordance with this data. In each iteration, the agent collects a batch of data by running the current policy in the environment. The collected data is used to update the policy by maximizing a surrogate objective function, which is designed to improve the policy while staying close to the original policy [32].

The principal concept underpinning PPO is the restriction of policy updates' magnitude to prevent excessive deviation of the new policy from the original one. This is attained by incorporating a penalty term into the objective function, which gauges the divergence between the new and original policies. This penalty term encourages small policy updates, which helps to prevent the agent from taking overly risky actions [31].

PPO has demonstrated its efficacy and efficiency as an algorithm for shaping policies in a multitude of settings, ranging from video games and robotics to finance [33], [34]. Its straightforwardness and scalability have rendered it a preferred option for numerous applications within the domain of RL.

2.4.12. Soft-Actor-Critic (SAC)

Soft Actor-Critic is an off-policy algorithm in DRL that combines ideas from both actor-critic methods and entropy regularization [35]. The actor-critic concept utilizes two models: the actor that decides which action to take, and the critic that evaluates the action taken by the actor. On the other hand, entropy regularization encourages the agent to explore a broader set of policies by assigning higher rewards to more random actions. By combining these two concepts, SAC is able to effectively learn a policy that is both performant and robust to variations in the environment.

The Soft Actor-Critic (SAC) algorithm is a popular RL algorithm that is used to solve continuous control problems. It belongs to the class of model-free, off-policy algorithms, which means that it does not require a model of the environment and can learn from any data, even if it was collected by a different policy [36].

Here's a brief overview of how the SAC algorithm works:

- **Actor-critic architecture:** The SAC algorithm uses an actor-critic architecture, which means that it learns both a policy (actor) and a value function (critic). The policy maps states to actions, and the value function estimates the expected return for each state-action pair.

- **Maximum entropy machine:** The SAC algorithm uses maximum entropy reinforcement learning, which means that it maximizes the expected reward while also maximizing the entropy of the policy. This encourages exploration and can lead to better performance.

- **Soft updates:** The SAC algorithm uses soft updates to update the target networks. This means that the target networks are updated slowly over time, which helps to stabilize the learning process.

- **Replay buffer:** The SAC algorithm uses a replay buffer to store and sample transitions. This allows the algorithm to learn from past experiences and can improve sample efficiency.

- **Exploration:** The SAC algorithm uses a technique called "action noise" to encourage exploration. This means that it adds noise to the actions selected by the policy, which can help the agent discover new and better policies.

2.4.13. Deep Deterministic Policy Gradient (DDPG)

Deep Deterministic Policy Gradient (DDPG) is an off-policy algorithm in DRL that marries the idea of deep Q-learning with policy gradients specifically tailored for continuous action spaces [37]. The structure of DDPG is dual-pronged, taking cues from actor-critic methods. The actor in DDPG maps states to actions deterministically, proposing the best-perceived action for a given state. Simultaneously, the critic evaluates the action proposed by the actor in the context of the Q-value function, guiding the actor's improvements. A unique trait of DDPG is its exploration strategy. Instead of the traditional epsilon-greedy methods used in discrete action spaces, DDPG introduces noise directly to its deterministic policy, typically via an Ornstein-Uhlenbeck process, promoting exploration in the continuous action landscape.

It falls under the category of model-free, off-policy algorithms, signifying that it neither requires a model of the environment for learning nor is it confined to learning from data generated by its own policy [37]. Here's a concise rundown of DDPG's key components and functionalities:

- **Actor-Critic Architecture**: DDPG employs an actor-critic paradigm. The actor takes on the role of the policy, mapping states to actions. The critic, conversely, is tasked with evaluating the value or quality of taking a particular action in a given state, usually in terms of the Q-value function.

- **Deterministic Policy Gradient**: Unlike traditional stochastic policies, DDPG operates on deterministic policies. The actor aims to output the most advantageous action for any given state directly, thus facilitating a focus on the essential task of optimizing policy performance.

- **Exploration via Noise Injection**: DDPG implements exploration strategies by adding noise to its deterministic policy, generally through an Ornstein-Uhlenbeck process. This approach helps the agent explore the continuous action space effectively, promoting the discovery of more optimal policies.

- **Replay Buffer**: Much like other off-policy algorithms, DDPG utilizes a replay buffer to store transitions of state, action, reward, and next state. This allows the algorithm to randomly sample and learn from its history, enhancing the efficiency of each sample.

- **Soft Updates**: To enhance stability in training, DDPG incorporates the idea of "soft" target updates. Instead of directly copying weights from the primary networks to target networks, DDPG employs a gentle blending of the weights, leading to smoother transitions and improved convergence.

Table 1. Comparison table between PPO, SAC and DDPG

Factor	PPO	SAC	DDPG
Sample Efficiency	Efficient	More efficient	Efficient
Policy Update	Small, incremental updates	Large, flexible updates	Deterministic policy with incremental updates
Exploration	Lower (due to on-policy nature)	Higher (due to entropy maximization)	Noise-based (Ornstein-Uhlenbeck process)
Implementation Complexity	Relatively simpler	More complex due to dual Q-functions and policy network	Moderate complexity with actor-critic policy
Training Stability	Stable due to clip factor	Stable due to soft updates and entropy regularization	Stable, especially with target networks
Sensitivity to Hyperparameters	Moderate sensitivity	Lower sensitivity, better generalization	Moderate sensitivity
Real-time Learning	Can handle real-time learning due to on-policy nature	Less suitable for real-time learning due to off-policy nature	Less suitable due to off-policy nature and reliance of replay buffer
Typical Applications	Wide range of general-purpose tasks	Tasks where high sample efficiency and exploration are needed	Continuous action space, task requiring a balance of exploration and exploitation

We created the following comparison table by combining the results from our modeling and literature information, see Table 1.

2.5. Related Work

In recent years, the application of ML has become increasingly common in the manufacturing industry. Traditional methods have started to fall short due to the rapid changes in production processes and requirements. At this juncture, the significance of structures developed with a data-driven approach, leveraging the power of data, has

grown. Consequently, the number of research studies conducted in this area is continually increasing.

Currently, industries involved in manufacturing are grappling with substantial obstacles like the need for mass customization and condensed product development cycles. Furthermore, amidst the nebulous and oscillating market demand, there is an exigency to conform to the continuously escalating expectations of product excellence and environmental steadiness within constrained timelines [38].

Nonetheless, these challenges concurrently pave the way for novel technologies to present fresh opportunities [39]. Let's first examine the literature from the perspective of DDS. For example, Rolls-Royce uses digital twins in its jet engine manufacturing process to analyze the condition of each component in real time and identify potential problems before they become severe and costly issues [40]. Also, some of the latest digital twins applications include using them to predict vehicle failures or detect and prevent product defects before they occur. These examples illustrate how digital twins can be used to improve manufacturing, maintenance, and safety processes in a wide range of industries. This helps the company minimize production downtime, reduce costs and improve customer satisfaction. Similarly, BASF has developed a digital twin of the gear unit used in its manufacturing plants to improve the operational efficiency of its production facilities and reduce maintenance-related downtime. The technology enables engineers to monitor the performance of the gear and compare it with data from similar units deployed across BASF plants. This enables them to identify inefficiencies and take remedial action to enhance the plant's overall performance [41], [42]. Several companies are turning to data analytics solutions that create use of AI and ML [42].

Continuous production systems generate and consume data continuously, as the name implies. These continuous flows lead businesses to monitor and control processes in real-time with the highest possible resolution. Before Industry 4.0 revolution and IoT devices, the most common concepts to satisfy these monitoring and control needs were simulations (especially discrete event simulation) created during the engineering and design phases of process equipment [43]. Simulation models could represent more or less all the design information but expecting real-world operational data is not applicable.

Therefore, they are not suitable for continuous process monitoring and control, and a bridge between this modeling discipline or further works/models/discipline became what businesses need. As compared to traditional methods, data-driven models tend to reflect real systems more accurately. It should be stated that real systems are capable of exhibiting behaviors that are not easily predicted and modeled in advance, and such behaviors can only be captured through the use of data [44]. Data-driven ML methods can be utilized to model phenomena that are difficult to reflect using standard simulation models. In addition, AI methods can be merged with simulation to gain a more profound knowledge of systems' behaviors [45], [46].

Within this scope, AI holds a distinctive position, and more specifically, DRL is progressively permeating the manufacturing sector, following the impressive outcomes exhibited by entities such as OpenAI [47] and DeepMind [48], among others. The critical characteristics of DRL, including its adaptability and capacity to generalize and learn from past experiences, have already impacted several fields, as indicated in various review studies. These include domains like robotics [49], scheduling [50], cyber-physical systems [51], and energy systems. In sectors that are intensive in resources and energy, like the fabrication of printed circuit boards (PCB), they showcased how a DRL algorithm could enhance lead time and cost efficiency while boosting revenues and diminishing carbon utilization, as compared to conventional approaches [52]. In a 2020 study, pitted a multi-agent DRL algorithm against a traditional mathematical modeling technique simulating the production of a lithium-ion battery [53]. The evaluation resulted in a 10% decrease in energy usage. Moreover, to achieve energy efficiency in hydrogen production, a DDS Environment (DDSE) based on ML was established, and through this environment, energy efficiency was attained [54].

To our best knowledge, this is the first study to simulate the hydrogen unit as a data-driven using ML techniques and to optimize hydrogen production by using this DDS as the environment of the DRL method.

CHAPTER 3

METHODOLOGY AND RESULTS

Hydrogen production is a major contributor to energy costs, and steam methane reforming is a commonly used method for producing hydrogen. This process involves converting methane gas and superheated steam into hydrogen, carbon dioxide, and carbon monoxide using a nickel-based catalyst at high temperatures. The reactions occur in tubes within the reformer furnace, which are heated by combustion within the furnace. The mixture of hydrogen, carbon monoxide, carbon dioxide, and unconverted methane produced by this process is sent to a purification unit where the hydrogen is separated from the other gases. The pure hydrogen is either used in the hydrocracker unit or sent to the hydrogen network, while the remaining mixture, known as tail gas, is used as fuel for the reformer furnace.

It is very difficult to carry out the production process efficiently, as hydrogen production is critical and costly. Many adjustable parameters can be changed in managing this process, and good changes can be observed by trial and error. However, changes and experiments in the real system can be costly and dangerous.

In this study, we focused on creating a novel approach to determine the set values that will make the unit run in the most optimal conditions and fully automized with ML and DRL algorithms in the H2 production unit. This can be considered as an optimization problem. However, unlike classical optimization solutions, we have established a completely novel data-centric approach. In this study, we propose a novel framework by combining two approaches which are data-driven digital twin via environment for the DRL and the DRL agent depicted in Figure 2. We have two stages in this framework, the first one is creating the digital twin simulation environment and the second one is creating DRL agent.

We offer field engineers a novel framework where they can simulate the real process and determine the best production situation. A digital twin is a model of an object or environment that can be used to simulate, understand, and improve its real-world counterpart through data analysis. AI, together with digital twin and simulation, can be used to significantly improve operational efficiency by modeling the impact of various operating parameters and enabling real-time adjustments to improve production efficiency and reduce operational costs. In addition, the digital twin and simulations help identify potential equipment failures early on, thereby reducing unplanned downtime and improving overall plant reliability. Additionally, this framework can be used to train field engineers.

At the first stage, we proposed a simulation-based data-driven digital twin framework by utilizing ML techniques. This framework is highly interactive, generalizable, and scalable compared to traditional simulation methods. It predicts how the changes made in the H2 production will affect the process in the future and easily adapts to the changes made in

the process thanks to its data-driven nature. In the second stage, in addition to these uses this framework will be used as the environment for DRL application.

In this study, ML models were trained using a dataset extracted from the process history database of the company. The company operates in the energy sector. It has a very high production rate and possesses data from numerous production sensors. The data has been collected from sensor of a unit that produces hydrogen. Throughout this process, we collaborated with chemical engineers working in the hydrogen production unit.

The dataset underwent preprocessing operations before being used to train the models. The trained models were subsequently utilized both to aid operators in controlling the process and in the optimization process.

In the second stage of this study, the optimization problem was formulated and subsequently solved. The process of formulating the optimization problem included the following steps:

• Determining the reward function

• Solving the optimization problem by using the most appropriate DRL algorithms.

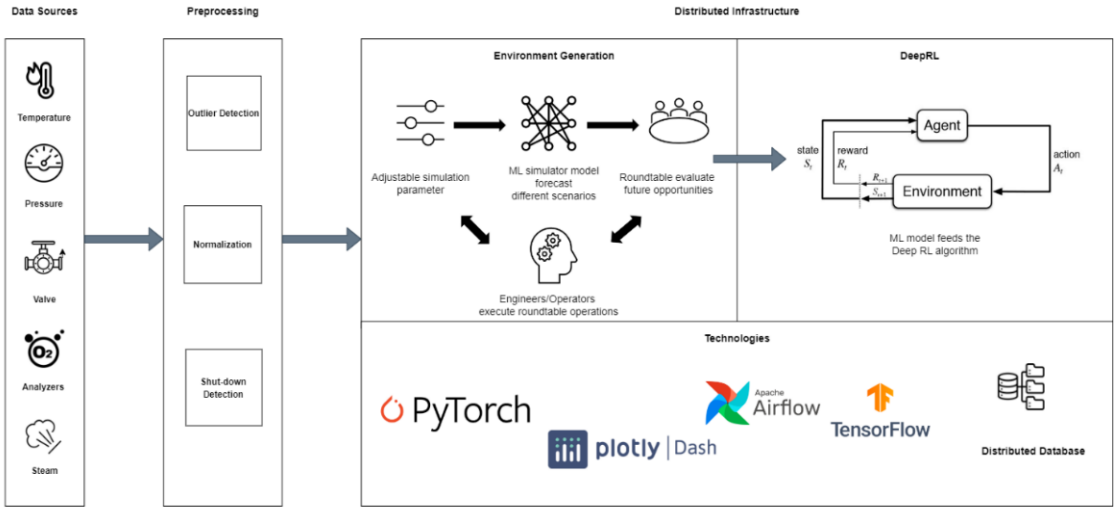

Figure 2. General framework

3.1. Data Sources

With the latest advancements in IIoT/IoT, we can collect structured, semi-structured, and unstructured data from diverse sets of manufacturing equipment, including temperature,

and pressure sensors, analyzers, and valve equipment from oil and gas refineries. The use of these IIoT/IoT technologies has significantly increased the efficiency of the manufacturing process by enabling manufacturers to continuously monitor and analyze the performance of various equipment used during the different stages of the production process. However, due to their heterogeneous nature, these data sources may generate data at a very high speed in disparate formats and units, such as Celsius and Kelvin values from steam pipes. Therefore, there is a need to address these variety and velocity challenges framework to collect, ingest and process these big data effectively. By taking into account the unique characteristics of today's big data, we utilize a resilient and fault-tolerant data distribution method to seamlessly transfer a large volume of data streams from data sources to distributed execution engine in a scalable and fault-tolerant manner. Apache Kafka, which is pioneered by LinkedIn. Sqoop, Pulsa, and RabbitMQ are well-known tools to define a loosely coupled architecture and ensure the scalability and reliability of the proposed infrastructure.

The utilization of ML models for the prediction of process values involves the incorporation of 18 set value sensor readings and twelve 5 min earlier process values and as input variables. You can see the feature names and their raw sampling rates in Table 2.

Table 2. Feature names and their sampling rates.

Feature Name	**Sampling Rate**	**Inputs&Targets**
West Furnace Draft	1 sample/30 seconds	Input 1
East Furnace Draft	1 sample/30 seconds	Input 2
Hydrocracker Charge Valve	1 sample/30 seconds	Input 3
Hydrocracker Charge Valve 2	1 sample/30 seconds	Input 4
Naphta	1 sample/30 seconds	Input 5
Reformer Gas	1 sample/1 minute	Input 6
Fuel Gas East	1 sample/30 seconds	Input 7
Fuel Gas West	1 sample/30 seconds	Input 8
Inlet Steam	1 sample/30 seconds	Input 9
E211 Outlet	1 sample/1 minute	Input 10
Reformer Outlet CH4	1 sample/1 minute	Input 11
E212 Koil Outlet	1 sample/1 minute	Input 12
E203 Outlet	1 sample/1 minute	Input 13
Density Anayzer Operation	1 sample/1 minute	Input 14
Inlet Steam 2	1 sample/30 seconds	Input 15
C205 Natural Gas	1 sample/30 seconds	Input 16
Recycle Hydrogen	1 sample/30 seconds	Input 17
Ratio	1 sample/1 minute	Input 18
Reformer Outlet CH4	1 sample/1 minute	Target 1
C Outlet CO	1 sample/1 minute	Target 2
PSA Inlet	1 sample/30 seconds	Target 3
Process Natural Gas Quantity	1 sample/30 seconds	Target 4

Table 2 cont.

F201 Flue Gas	**1 sample/1 minute**	**Target 5**
K203 Outlet Steam	1 sample/30 seconds	Target 6
Reformer East Outlet	1 sample/1 minute	Target 7
Reformer West Outlet	1 sample/1 minute	Target 8
PSA H2 Outlet	1 sample/30 seconds	Target 9
PSA Outlet Tail Gas	1 sample/30 seconds	Target 10
C9 Drum Pressure Control	1 sample/30 seconds	Target 11
East Furnace Draft	1 sample/30 seconds	Target 12
West Furnace Draft	1 sample/30 seconds	Target 13
Hydrogen From C9 to Fuel Gas	1 sample/30 seconds	Target 14
East Outlet Maximum Temprature	1 sample/30 seconds	Target 15
West Outlet Maximum Temprature	1 sample/30 seconds	Target 16

Upon examining the processes and the characteristics of the available data, it was determined that data corresponding to five-minute intervals would be utilized in the analysis. When determining this interval, consultations were held with field engineers. It was indicated that the most optimal value during the usage would be 5 minutes and viewing the data at 5-minute resolution would be sufficient. Furthermore, the flow for the proposed DDS is delaminated in Figure 3.

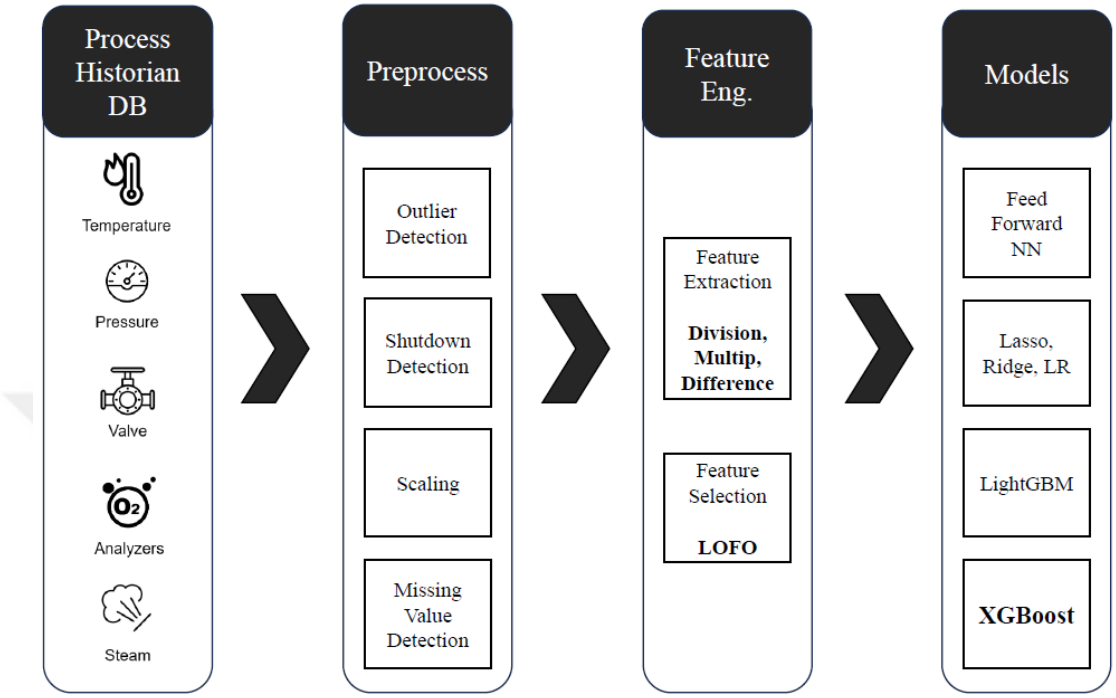

Figure 3. Data-driven simulation structure flow.

3.2. Data Preprocessing

Data preprocessing plays a crucial role in developing a successful AI application. This module aims to prepare data for modeling and to generate a digital-twin environment. It aims to prepare data by removing outliers from data, normalizing data to reduce complexity and redundancy in the data set, and also detecting shutdowns and removing these data from the data set to improve the success of the digital twin model.

The data preprocessing module also converts data into a common format for further processing and determining the optimum lag time to understand how long the changes in adjustable parameters affect the process. Moreover, this module also focuses on determining the features that best describe the efficiency of the production unit by using AI algorithms and crosschecking them with domain experts, including process and chemical engineers. Moreover, the most critical inputs and outputs for DDS were selected by manufacturing professionals and using ML-based feature selection techniques. The process of performing analytics on streaming data requires the availability of data sources, the capability to scale to high volumes, and the assurance of high-quality data.

3.2.1. Outlier and Shutdown Elimination

The energy production plants are not structures that operate continuously. There may be breakdowns and planned maintenance shutdowns. In this case, it is necessary to identify and clean these shutdowns from our raw data set. We identified and cleaned these shutdowns both from the data itself and with the help of domain experts.

Later, we used both field knowledge and outlier detection methods to clean the outliers from the data. At this point, we identified the outliers using local outlier factor analysis, and progressed by cross-verifying with field knowledge, eliminating the outliers in the data. The Local Outlier Factor (LOF) [55] is a technique employed to detect unusual data elements by determining the local divergence of a particular data element in relation to its surrounding elements. This is achieved by computing the ratio between the density of a data element's neighbors and the density of the overall data collection. This technique proves to be especially advantageous in pinpointing local outliers, anomalies that appear to be unusual primarily in comparison to their immediate neighbors, as opposed to the whole data set.

3.2.2. Missing Value

Our data set is located in the company's process historian database, so we did not encounter any missing values because the system is robust. Additionally, we did not experience any data loss because we used Kafka for data ingestion.

3.2.3. Scaling

The min-max scaler is useful for datasets where some features may have significantly larger values than others, as it can help to normalize the data and make the features comparable. Within the framework of a dataset pertaining the utility of a min-max scaler comes into play when making comparisons between features, which might have different units or scales. This could include parameters like the temperature of a chemical reaction or the pressure within a pipeline. The process of scaling the data in this manner enables the detection of data trends or patterns that may not be overtly discernible otherwise. We applied min-max scaling to the entire dataset used in modelling, scaling values between 0 and 1, since the models will be used to create the environment and Deep RL will be run on it.

Following the successful completion of our data preprocessing steps, we are then poised to transition to the phase of model construction.

3.2.4. Feature Extraction

As part of the data preprocessing, we applied feature extraction techniques to generate new features from the 18 original input variables. The goal of this process was to derive features that could potentially improve the performance of our models, by capturing additional information about the relationships between the original variables. To this end, we included multiplications and powers of the input features. In this extraction process, we included both polynomial features and interaction terms up to the second order. Furthermore, to obtain the division-based interactions, we applied polynomial feature extraction with degrees of -1 and 1. Also, we extracted the rate of change for each input feature by taking the difference with its subsequent time point. The feature extraction methodology was implemented across entire input set.

The resulting feature set contained the original 18 input variables as well as the new features generated from the above operations. With these feature extraction techniques which are mentioned above, we obtained 347 features in total. It's essential to recognize that these transformations, while expanding the dimensionality of our feature landscape, also furnished our models with extra data that could be utilized to decipher the intricate connections embedded in our dataset. To safeguard our models from being swamped by potentially inconsequential features, we implemented suitable feature selection methods (elaborated in the subsequent section) to determine the features that held the most significance in relation to our target variable.

3.2.5. Feature Selection

After the feature extraction stage, our dataset contained more than 300 features. This increased dimensionality, while providing our models with more potential sources of information, also risked introducing noise and overfitting. To ensure that our models focused on the most relevant features, we used feature selection to identify and keep only the features that contributed most significantly to our model's predictive performance.

In our study, we used a multi-model approach, where each of our 16 models given in Section 3.3 was designed to predict a different target variable. While all models shared the same original pool of 18 input variables, the relevance of these inputs could vary significantly across the different targets. Therefore, it was necessary to perform feature extraction and selection individually for each model, allowing

each one to focus on the subset of features most relevant to its specific target variable.

For feature selection, we used the Leave One Feature Out (LOFO) [56] importance method. LOFO, a model-independent and intuitive methodology, quantifies the significance of a feature by evaluating the influence on the model's performance when that specific feature is eliminated.

The computation of a feature's importance by LOFO involves multiple training iterations of the model, in each of which a different feature is excluded from the training set. Subsequently, the model's performance (including accuracy, F1-score, and so on) is compared with and without the inclusion of the feature. The decline in performance metrics underlines the importance of the excluded feature: a pronounced decrease suggests the essential nature of the feature, while a slight decrease implies that the feature could be of little relevance or even adverse.

This approach provides a notable advantage in assessing the aggregate influence of each feature in relation to the model, factoring in any potential interactions among the features. Unlike some other feature importance methods, LOFO does not assume that features contribute independently to the model's performance, making it well-suited to datasets with complex, non-linear relationships and interactions.

LOFO also provides us a feature importance score for each feature. This allows us to prune less important ones and keep the model focused on the most informative features. In feature selection, features obtained through the division technique generally is emphasized. The selected number of features varies for each target. The number of features selected for each target can be seen in Figure 4. The selected features were then used for training our final models, helping us to achieve robust and interpretable results with a manageable level of complexity.

After feature extraction, utilized the LOFO feature importance scores to rank all possible features by their importance for each model. For each target variable, the model was trained multiple times, each time leaving out a different feature from the training set, and the impact on the model's performance was noted. The features were then ranked based on the drop in performance metrics when they were left out, and the top-ranking features were selected for the final model. The final selected features are presented in Appendix B.

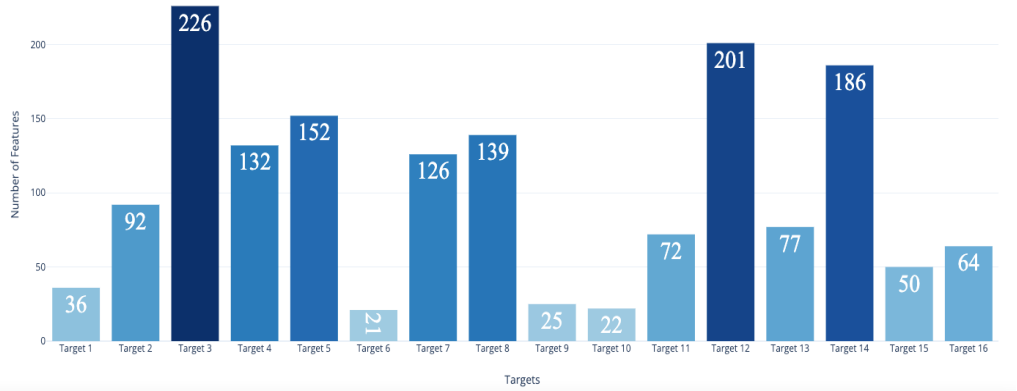

Figure 4. The selected number of features for each target.

3.3. Multiple Model Training

In our study, we developed 16 different predictive models, each corresponding to a different target variable. Despite these different targets, all models were built starting from the same original set of 18 input variables. This allowed us to explore and compare how different input features influence different targets in the context of our specific problem domain.

This approach enables us:

i. Ensuring that developed analytical models do not use irrelevant or potentially noisy features, thereby reducing overfitting potential and improving the interpretability of analytical models for each target.

ii. Understanding the importance of different input features varied across different target variables. Some features, for example, may be very important for predicting one target but less important for others. By applying feature selection individually for each model, we could capture these differences and tailor each model to its specific target. Modeling this unit was not easy because the relationships and objectives of the target variables were different. Therefore, we had to consider many features and examine their relationships.

The selected features for each model, while derived from the same original set, were therefore potentially different for each target variable. This allowed us to achieve optimal performance on each target while also gaining insights into how different features influence different targets.

3.3.1. Modelling Part

In this section, we present a methodology utilizing ML techniques for the purpose of analyzing the given data. Initially, we elaborate on the model architecture employed in our analysis. Following this, we outline the criteria used for assessing the effectiveness of the models, aiming to compare the competencies of various models.

We conducted training sessions with a diverse set of ML models using the data, with the intent of contrasting their performance. These models include Linear, Lasso, Ridge Regression XGBoost, LightGBM and Feed Forward Neural Network algorithms. Moreover, to ascertain the solidity of our outcomes, we implemented 5-fold cross-validation as a strategy to verify our model. Our initial approach was based on time series. The results have been added to Appendix A. Later, we consulted with the field engineers and incorporated their domain knowledge into our work. We shared our findings with them and engaged in brainstorming discussions. Based on their feedback, we understood that the system was more influenced by the impact. Armed with this knowledge, we updated our model and used a random split. To identify which algorithm provides the best alignment with our data, we performed tests using six distinct foundational models.

• **Base Model Comparison:** At the initial stage, we conducted a set of experiments using a selection of well-established ML algorithms to build a base model for each. The algorithms tested included Linear Regression, Lasso Regression, Ridge Regression, XGBoost, LightGBM, and a FFNN. Each model was implemented using their respective default hyperparameters which is provided in Table 3, providing a fair basis for the initial comparison. The models were evaluated and compared based on RMSE and RMSE/STD which means Root Relative Squared Error (RRSE). Also, the details of RMSE can be found in Equation 2, where $\hat{y}_i$ represents the predicted values and y_i represents the real target values. RRSE indicates how well the model performs compared to a naive model that always predicts the mean of the target values. In Equation 3, RRSE explained in detail where n is sample size, $\hat{y}_i$ is predicted value, $\bar{y}$ is average of real values and y_i is the real value. An RRSE value greater than 1 suggests that the model's predictions are worse than a model that predicts the mean, while values approaching 0 indicate increasingly accurate predictions. Lower RMSE values indicate better performance. RMSE/STD offers a standardized version of RMSE. The lower the value, the better the performance. The evaluation was carried out separately for each of the 16 targets to provide a thorough understanding of each model's performance across all problem areas.

$$RMSE(Y) = \sum_{i=0}^{n} \frac{(\hat{y}_i - y_i)^2}{n} \qquad (2)$$

$$RRSE(Y) = \sqrt{\frac{\sum_{i=0}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{n}(y_i - \bar{y})^2}} \quad (3)$$

Table 3. Base models default hyperparameters

Parameter/ Model	**Linear Reg.**	**Lasso Reg.**	**Ridge Reg.**	**XGBoost**	**LightGBM**	**FFNN**
Library	sklearn	sklearn	sklearn	xgboost	lightgbm	tensorflow
fit_intercept	True	True	True	N/A	N/A	N/A
normalize	False	False	False	N/A	N/A	N/A
alpha	N/A	1.0	1.0	N/A	N/A	N/A
max_iter	N/A	1000	None	N/A	N/A	N/A
tol	N/A	0.0001	0.001	N/A	N/A	N/A
booster	N/A	N/A	N/A	gbtree	N/A	N/A
Objective	N/A	N/A	N/A	reg:squarederror	regression	Mean Squared Error
learning_rate	N/A	N/A	N/A	0.3	0.1	0.001
N_estimators	N/A	N/A	N/A	100	100	N/A
Max_depth	N/A	N/A	N/A	6	-1(no limit)	N/A

Table 3 cont.

Min_child_weight	N/A	N/A	N/A	1	0.001	N/A
subsample	N/A	N/A	N/A	1	1.0	N/A
Colsample_bytree	N/A	N/A	N/A	1	1.0	N/A
Lambda(L2 reg.)	N/A	N/A	N/A	1	0	N/A
Alpha(L1 reg.)	N/A	N/A	N/A	0	0	N/A
dropout	N/A	N/A	N/A	N/A	N/A	0.1
Hidden_layers	N/A	N/A	N/A	N/A	N/A	2 layers, 200 neurons for each layer
optimizer	N/A	N/A	N/A	N/A	N/A	Adam

As seen in the model results in Table 4, models that can capture non-linear and more complex relationships gave better results for our 16 target variables. Here we can say that there is a complex and nonlinear relationship between the sensors. According to the test results, XGBoost algorithm outperforms the other algorithms in RMSE and RMSE/STD scores, thus it is selected as the base model. This is an expected result, as in studies conducted for tabular data, it has been observed that XGBoost is more prominent [57].

Upon detailed examination for the 16 targets, the average RMSE/STD value was found to be 0.231 for XGBoost, 0.255 for LGBM, 0.316 for FFNN, and approximately 0.42 for Linear, Lasso and Ridge regression models. Notably, XGBoost performed considerably better, especially when compared to linear regression-based models.

In addition, the feed forward neural network may require more in-depth and time-consuming tuning. We conducted these experiments using the original features, without applying any feature extraction or selection techniques. Table 5 shows details how the model results change after applying feature extraction and selection methods.

Table 4. Models' train and test results in terms of RMSE and RMSE/STD

Models		**Linear**		**Lasso**		**Ridge**		**XGBoost**		**LightGBM**		**FFNN**	
		RMSE	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**
Target 1	Train	0.2468	0.871	0.2624	0.925	0.2468	0.871	**0.114**	**0.402**	0.147	0.519	0.201	0.711
	Test	0.2461	0.868	0.2614	0.922	0.2461	0.868	**0.146**	**0.517**	0.156	0.051	0.201	0.712
Target 2	Train	0.242	0.554	0.297	0.678	1447.76	0.128	**0.04**	**0.09**	0.05	0.116	0.107	0.245
	Test	0.233	0.532	0.289	0.661	1428.27	0.126	**0.056**	**0.12**	0.06	0.138	0.112	0.257
Target 3	Train	1447.76	0.128	1459.62	0.129	1447.76	0.128	**456.97**	**0.04**	575.18	0.051	890.67	0.079
	Test	1428.26	0.126	1432.87	0.127	1428.27	0.126	**529.49**	**0.047**	581.69	0.051	890.27	0.079
Target 4	Train	597.3	0.091	603.01	0.092	597.3	0.091	**90.61**	**0.013**	138.13	0.021	285.11	0.043
	Test	595.59	0.091	598.22	0.091	595.59	0.091	**123.82**	**0.019**	147.83	0.022	284.12	0.043
Target 5	Train	0.922	0.086	1.17	0.109	0.922	0.086	**0.28**	**0.026**	0.383	0.036	0.554	0.052
	Test	0.911	0.085	1.16	0.109	0.911	0.085	**0.343**	**0.03**	0.402	0.037	0.561	0.052
Target 6	Train	691.09	0.075	708.58	0.077	691.09	0.075	**320.02**	**0.035**	406.73	0.044	480.67	0.052
	Test	669.86	0.073	674.07	0.074	669.86	0.073	**393.68**	**0.043**	431.87	0.047	491.36	0.053
Target 7	Train	4.732	0.242	5.045	0.258	4.732	0.242	**1.580**	**0.081**	2.049	0.105	2.611	0.134

Table 4 cont.

	Test	4.70	0.241	4.994	0.256	4.705	0.241	**1.913**	**0.098**	2.166	0.111	2.913	0.149
Target 8	Train	4.44	0.227	4.44	0.227	4.44	0.227	**1.644**	**0.084**	2.150	0.110	2.847	0.145
	Test	4.38	0.224	4.38	0.224	4.38	0.224	**1.923**	**0.098**	2.182	0.111	3.003	0.153
Target 9	Train	1069.64	0.160	1074.32	0.160	1069.64	0.160	**364.28**	**0.054**	465.24	0.069	510.67	0.076
	Test	1050.84	0.157	1053.10	0.157	1050.84	0.157	**434.21**	**0.065**	470.39	0.070	531.42	0.079
Target 10	Train	576.94	0.131	582.60	0.132	576.94	0.131	**219.68**	**0.049**	272.97	0.062	301.17	0.068
	Test	576.45	0.130	578.22	0.131	576.45	0.130	**262.99**	**0.059**	284.89	0.064	319.54	0.072
Target 11	Train	4.989	0.959	5.059	0.972	4.989	0.959	**3.226**	**0.620**	3.864	0.742	4.743	0.911
	Test	5.032	0.967	5.085	0.977	5.032	0.967	**3.838**	**0.737**	4.091	0.786	5.057	0.972
Target 12	Train	0.666	0.929	0.698	0.973	0.666	0.929	**0.316**	**0.441**	0.413	0.577	0.491	0.684
	Test	0.664	0.927	0.697	0.972	0.664	0.927	**0.365**	**0.509**	0.429	0.598	0.534	0.744
Target 13	Train	0.548	0.920	0.582	0.977	0.548	0.920	**0.212**	**0.356**	0.287	0.482	0.316	0.530
	Test	0.548	0.920	0.582	0.976	0.548	0.920	**0.261**	**0.438**	0.302	0.508	0.339	0.568
Target 14	Train	887.35	0.935	887.54	0.935	887.35	0.935	**583.94**	**0.615**	691.11	0.728	724.21	0.763
	Test	886.22	0.934	886.22	0.934	886.22	0.934	**677.12**	**0.713**	713.59	0.752	785.75	0.828

Table 4 cont.

Target 15	Train	5.088	0.258	5.353	0.271	5.088	0.258	**1.679**	**0.085**	2.210	0.112	2.624	0.133
	Test	5.066	0.257	5.309	0.269	5.066	0.257	**2.021**	**0.102**	2.327	0.118	2.953	0.149
Target 16	Train	4.617	0.237	4.869	0.250	4.617	0.237	**1.672**	**0.085**	2.173	0.111	2.572	0.132
	Test	4.623	0.237	4.875	0.250	4.623	0.237	**2.003**	**0.102**	2.286	0.117	2.847	0.146

Table 5. Model results with feature engineering

Models		Linear		Lasso		Ridge		XGBoost		LightGBM		FFNN	
		RMSE	**RMSE/ STD**	**RMSE**	**RMSE/ STD**	**RMSE**	**RMSE/ STD**	**RMSE**	**RMSE/ STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**
Target 1	**Train**	0.2823	0.996	0.2318	0.818	0.1947	0.687	**0.090**	**0.319**	0.126	0.445	0.211	0.746
	Test	0.2887	1.018	0.2387	0.842	0.2085	0.735	**0.132**	**0.467**	0.146	0.516	0.215	0.759
Target 2	**Train**	0.468	1.070	0.219	0.501	0.184	0.422	**0.035**	**0.08**	0.045	0.103	0.117	0.267
	Test	0.467	1.067	0.211	0.483	0.181	0.415	**0.050**	**0.11**	0.051	0.118	0.132	0.301
Target 3	**Train**	1657.45	0.147	945.32	0.083	771.24	0.068	**413.79**	**0.036**	519.92	0.046	810.41	0.071
	Test	1688.14	0.149	941.39	0.083	773.65	0.068	**524.08**	**0.046**	554.59	0.049	870.82	0.077
Target 4	**Train**	510.24	0.078	348.63	0.053	285.12	0.043	**83.50**	**0.012**	117.97	0.018	252.37	0.038
	Test	525.17	0.080	343.07	0.052	282.40	0.043	**123.54**	**0.018**	137.65	0.021	255.71	0.039
Target 5	**Train**	0.801	0.075	0.757	0.071	0.599	0.056	**0.252**	**0.023**	0.330	0.031	0.384	0.036
	Test	0.817	0.076	0.745	0.069	0.610	0.057	**0.321**	**0.030**	0.354	0.033	0.392	0.036
Target 6	**Train**	496.21	0.054	495.92	0.054	428.83	0.047	**278.06**	**0.030**	360.82	0.039	391.41	0.042
	Test	500.32	0.054	497.10	0.054	431.14	0.047	**372.01**	**0.040**	401.42	0.044	405.38	0.044
Target 7	**Train**	3.471	0.178	3.288	0.168	2.693	0.138	**1.298**	**0.066**	1.719	0.088	2.365	0.121

Table 5 cont.

	Test	3.526	0.180	3.342	0.171	2.772	0.142	**1.785**	**0.091**	1.926	0.098	2.489	0.127
Target 8	**Train**	3.547	0.181	3.382	0.173	2.768	0.141	**1.366**	**0.069**	1.822	0.093	2.267	0.115
	Test	3.624	0.185	3.425	0.175	2.834	0.144	**1.872**	**0.095**	2.039	0.104	2.313	0.118
Target 9	**Train**	700.84	0.104	683.44	0.102	582.05	0.087	**303.89**	**0.045**	407.39	0.061	476.37	0.071
	Test	711.68	0.106	687.07	0.102	587.07	0.087	**410.67**	**0.061**	437.15	0.065	496.78	0.074
Target 10	**Train**	405.94	0.092	396.96	0.090	343.52	0.078	**186.97**	**0.042**	235.76	0.053	280.41	0.063
	Test	410.27	0.093	396.09	0.090	346.03	0.078	**250.22**	**0.056**	258.83	0.058	295.67	0.067
Target 11	**Train**	4.910	0.943	4.909	0.943	4.596	0.883	**2.729**	**0.524**	3.501	0.673	4.001	0.769
	Test	5.001	0.961	4.935	0.948	4.646	0.893	**3.618**	**0.695**	3.797	0.730	4.198	0.807
Target 12	**Train**	0.636	0.886	0.624	0.870	0.559	0.780	**0.287**	**0.401**	0.382	0.533	0.417	0.581
	Test	0.651	0.907	0.618	0.862	0.588	0.820	**0.340**	**0.474**	0.397	0.553	0.424	0.591
Target 13	**Train**	0.531	0.890	0.522	0.876	0.455	0.764	**0.189**	**0.317**	0.261	0.439	0.291	0.488
	Test	0.542	0.909	0.518	0.870	0.457	0.767	**0.244**	**0.409**	0.282	0.473	0.302	0.506
Target 14	**Train**	887.35	0.935	887.54	0.935	887.35	0.935	**583.94**	**0.615**	691.11	0.728	724.21	0.763
	Test	886.22	0.934	886.22	0.934	886.22	0.934	**677.12**	**0.713**	713.59	0.752	785.75	0.828

Table 5 cont.

Target 15	**Train**	5.088	0.258	3.692	0.187	3.093	0.157	**1.396**	**0.070**	1.898	0.096	2.031	0.103
	Test	5.066	0.257	3.739	0.189	3.199	0.162	**1.880**	**0.095**	2.093	0.106	2.296	0.116
Target 16	**Train**	3.726	0.191	3.526	0.181	2.895	0.148	**1.424**	**0.073**	1.868	0.096	2.133	0.109
	Test	3.831	0.196	3.593	0.184	2.976	0.152	**1.908**	**0.098**	2.097	0.107	2.287	0.117

After implementing feature engineering, we observed significant improvements, especially in linear-based models. We also noted enhancements in non-linear based models. XGBoost consistently outperforms other models across most targets for both training and test datasets. This indicates that XGBoost is generally the best model for this specific dataset and problem.

In more detailed analysis of the results, the engineered features yielded an average of 16 targets test RMSE/STD values 0.448, 0.381 and 0.346 for Linear, LASSO and RIDGE regression models, respectively. Especially after feature engineering in lasso and ridge regression, it has been observed that the RMSE/STD value decreased from 0.42 to 0.38 and 0.35, respectively. On the test dataset, the Ridge model seems to generalize better than the Lasso model despite having a slightly higher RMSE in the training dataset. XGBoost and LGBM seem to have increased RMSE values on the test data compared to the train data, indicating possible overfitting. The XGBoost, LGBM and FFNN models produced average 16 targets test RMSE/STD values of 0.218, 0.239 and 0.287, respectively. Notably, XGBoost stood out with the engineered feature set. The ability of XGBoost to provide both accurate and swift results present an apt fit for our problem. For this reason, we chose the XGBoost model for hyperparameter tuning.

- **Model Optimization via Hyperparameter Tuning:** Following the base model comparison, the model yielding the best performance was selected for further optimization. Hyperparameter tuning was conducted to improve the performance of the selected model, XGBoost. Hyperparameters can be defined as model parameters that are preset before training, and the optimization of these values can have a substantial impact on model performance. The process involves exploring a range of potential parameter values and determining the combination that produces the most accurate predictions on our data. We used Optuna library in Python to tune the XGBoost models. Optuna is a hyperparameter optimization framework in Python. Its primary goal is to facilitate the optimization process of ML model parameters in more efficient and automated way. Optuna aims to find the optimal hyperparameters for your model based on the user-defined objective function. To prevent model overfitting, we crafted our objective function by adding the absolute difference between training RMSE and validation RMSE to the validation RMSE, see Equation 4. By doing so, we sought to mitigate the selection of models exhibiting significant discrepancies between training and validation, thereby preempting potential overfitting issues.

$$f_{obj} = RMSE(Y_{valid}) + |RMSE(Y_{train}) - RMSE(Y_{valid})| \qquad (4)$$

In the below, we explained the hyperparameters which is searched by the Optuna's hyperparameter search space in detail. Furthermore, you can see the hyperparameter names and their optimal values for each 16 targets in Table 6.

- **learning_rate (or eta)**:

  It determines the step size at each iteration while moving towards a minimum of the loss function. Its range is between 0 and 1. Smaller values can make the optimization more robust but require more boosting rounds. A too large learning rate may lead to convergence to suboptimal solutions.

- **max_depth**:

  It determines the maximum depth of a tree. It varies commonly between 3 and 10. Deeper trees capture more complex patterns but can also overfit. A smaller value can prevent overfitting but may be too simplistic to capture underlying patterns.

- **subsample**:

  Subsample is a fraction of the training samples used to train each tree. Its range is between 0 and 1. A smaller value prevents overfitting but can make the algorithm too conservative. It introduces randomness to make the model robust to noise.

- **colsample_bytree (or colsample_bylevel, colsample_bynode)**:

  It defines fraction of features to be randomly sampled for building each tree. Its range is between 0 and 1. Similar to subsample, introduces randomness to the feature selection to make models more robust and prevent overfitting.

- **L1 regularization (alpha)**:

  It is a L1 regularization term on weights, leading to feature selection. It takes non-negative values as input. Larger values make the algorithm more conservative. L1 can lead to sparsity, which means some feature weights can become exactly zero, effectively making the model simpler.

- **L2 regularization (lambda)**:

  It is L2 regularization term on weights. It takes non-negative values as input as L1 regularization. Larger values make the algorithm more conservative. L2 tends to prevent extreme weight values but doesn't lead to sparsity.

- **n_estimators**:

  It is the number of boosting rounds or trees to be trained. The input value depends on the dataset and problem. Too many estimators can lead to overfitting. Usually combined with early stopping to prevent overfitting.

- **gamma (or min_split_loss)**:

  Gamma value specifies the minimum loss reduction required to make a further partition on a leaf node. It accepts non-negative values as input. Larger values make the algorithm conservative by making it more stringent about the reduction in the loss.

- **min_child_weight**:

  This hyperparameter is the minimum sum of instance weight (hessian) needed in a child. The input is a non-negative value. It defines the minimum sum of samples (weights) in a child. If a partition results in leaves with a sum less than min_child_weight, then the partitioning step is not done.

After completing the hyperparameter tuning, we observe a notable improvement in the test results. Additionally, compared to the base model, the train and test errors have converged, indicating that through tuning, we achieved models with better generalization.

Table 6. The optimal hyperparameter setups for 16 different targets (All of them are XGBoost models)

Targets	Learning Rate	Max Depth	Number of Estimators	L1 reg	L2 reg	Subsample	Colsample bytree	Min Child	Gamma	Train RMSE	Test RMSE	Train RMSE/ STD	Test RMSE/ STD
Target 1	0.062	4	948	2.09e-05	2.70e-05	0.575	0.492	6	0.01	**0.104**	**0.131**	**0.369**	**0.463**
Target 2	0.025	6	990	0.014	0.0011	0.870	0.10	8	1.326e-05	**0.037**	**0.047**	**0.084**	**0.107**
Target 3	0.06	5	982	0.001	0.0008	0.737	0.188	3	0.0003	**378.51**	**466.73**	**0.033**	**0.041**
Target 4	0.021	7	706	0.153	2.35e-05	0.697	0.158	6	0.0129	**76.046**	**105.96**	**0.011**	**0.016**
Target 5	0.035	8	578	0.0003	6.005e-05	0.695	0.686	8	3.36e-05	**0.161**	**0.237**	**0.015**	**0.022**
Target 6	0.028	6	726	0.144	9.12e-05	0.769	0.457	6	2.59e-05	**277.99**	**339.42**	**0.030**	**0.037**
Target 7	0.193	4	965	9.189e-05	0.030	0.537	0.185	8	0.024	**1.075**	**1.503**	**0.055**	**0.077**
Target 8	0.175	4	614	0.0005	9.548	0.976	0.275	4	0.114	**1.313**	**1.706**	**0.067**	**0.087**
Target 9	0.064	4	799	0.0006	0.956	0.818	0.415	5	0.035	**348.54**	**398.37**	**0.052**	**0.059**
Target 10	0.022	7	460	1.151e-05	0.0162	0.839	0.8109	5	0.0629	**191.27**	**236.00**	**0.043**	**0.053**
Target 11	0.272	3	416	1.406	6.26e-05	0.921	0.713	6	7.104e-05	**3.171**	**3.602**	**0.609**	**0.692**

Table 6 cont.

Target 12	0.120	4	941	1.653	0.006	0.581	0.266	5	0.00023	**0.279**	**0.321**	**0.389**	**0.448**
Target 13	0.016	10	772	0.0007	0.156	0.792	0.543	8	0.318	**0.166**	**0.220**	**0.279**	**0.370**
Target 14	0.174	3	548	1.906	0.00033	0.592	0.348	4	2.339	**581.15**	**631.95**	**0.612**	**0.666**
Target 15	0.165	5	553	0.199	8.264	0.670	0.986	2	5.818	**1.135**	**1.626**	**0.057**	**0.082**
Target 16	0.180	4	496	3.88e-05	0.047	0.672	0.119	8	0.262	**1.481**	**1.809**	**0.076**	**0.092**

3.4. Deep Reinforcement Learning Models

In the previous stage, the developed DDS model will be used for the DRL study. At this point, we predict the environment that will occur in the next step based on the agent's action using the model we created. The success of our environment model will also affect the success of our DRL model. In the domain of deep reinforcement learning (RL), our primary objective is to optimize the C9 drum pressure. The value of the C9 drum pressure affects both the waste hydrogen value and the consumed natural gas value. At this point, in the initial phase, we utilized the C9 drum pressure model from the data-driven simulation environment we had developed. The environment we are working on consists entirely of continuous data, so we applied the PPO, SAC and DDPG Deep RL algorithms. In the following, the reward, the logic of the PPO and SAC algorithms are explained and discussed with the results. DRL models' parameters, and their descriptions can be seen in Table 7.

Table 7. DRL models' parameters

Algorithm	**Hyperparameter**	**Value**	**Description**
PPO	Learning Rate	0.0003	Controls the step size during optimization
	Clip Parameter	0.2	Clips the policy ratio to prevent excessively large policy updates
	Number of Epochs	10	Number of times the algorithm iterates over the collected data during each update
	Gamma	0.99	Factor by which future rewards are discounted
	Gae Lambda	0.95	Trading of bias vs variance
	Batch Size	64	Mini batch size
	Actor Network Layers	2	Number of layers for actor network
	Actor Network Units for each Layer	64	Number of neurons for each layer
	Critic Network Layers	2	Number of layers for critic network

Table 7 cont.

	Critic Network Units for each layer	64	Number of neurons for each layer
	Activation functions	“relu”	Activation functions for actor and critic network
SAC	Learning Rate	0.0003	Controls the step size during optimization
	Buffer Size	1000000	Size of the replay buffer
	Learning Starts	100	Number of initial time steps taken by the model to accumulate transition data prior to commencing the learning process.
	Batch Size	256	Mini batch size for each gradient update
	Tau	0.005	Soft update coefficient
	Gamma	0.99	Factor by which future rewards are discounted
	Entropy Coefficient	“auto”	Entropy regularization parameter, it controls the exploration/exploitation trade off.
	Target Update İnterval	1	Update target network every frequency gradient steps
	Actor Network Layers	2	Number of layers for actor network
	Actor Network Units for each Layer	256	Number of neurons for each layer
	Critic Network Layers	2	Number of layers for critic network
	Critic Network Units for each layer	256	Number of neurons for each layer
	Activation functions	“tanh”	Activation functions for actor and critic network

Table 7 cont.

DDPG	Actor Network Learning Rate	0.0001	Actor network learning rate
	Critic Learning Rate	0.0001	Critic Network Learning rate
	Gamma	0.99	Factor by which future rewards are discounted
	Tau	0.001	Soft update parameter
	Batch Size	128	Batch size for learning the policy
	Buffer Size	50000	Size of the replay buffer
	Action Noise	OrnsteinUhlenbeck Action Noise	The action noise type
	Actor Network Layers	2	Number of layers for actor network
	Actor Network Units for each Layer	400	Number of neurons for each layer
	Critic Network Layers	2	Number of layers for critic network
	Critic Network Units for each layer	300	Number of neurons for each layer
	Activation functions	“tanh”	Activation functions for actor and critic network

3.4.1. Reward

In DRL, the design of the reward function plays a crucial role in guiding the agent's learning process. This section focuses on the development of a reward function for minimizing sensor values in the context of our thesis. We begin by providing an overview of the problem and its relevance to the sensor data. Subsequently, we discussed the formulation of the reward function that effectively drives the agent to achieve the desired objective. Our study aims to minimize the values of specific sensors, which is the amount of hydrogen released into the fuel gas, i.e., the excess hydrogen produced and considered

waste in each system using DRL. The minimization of the excess hydrogen values is essential for optimizing system performance, reducing the cost.

The main objective of this optimization problem is stabilizing the hydrogen pressure sensor at 60 bar. According to domain knowledge, if the system pressure is maintained consistently at 60 bar, the amount of hydrogen wasted is minimized, ensuring the system operates in a stable manner. At this juncture, we established our objective to set the system pressure to 60 bar. Accordingly, we defined our reward function, f_{reward}, see Equation 4, where P is the pressure value. The reward function was predicated on the system pressure reaching 60 bar.

$$f_{reward} = -|60 - P| \tag{4}$$

3.5. Results

The framework we present in this thesis facilitates reducing production costs, helping professional field engineers, and training new incoming field engineers. While planning to evaluate the outcomes, it became apparent that an extended testing duration was necessary. This period was crucial to gauge the framework's effectiveness in disseminating knowledge to subsequent generations and accurately determining the ideal setup for the best production. When tests were made with historical data, it was observed that the amount of wasted hydrogen decreased.

In our case, we could not obtain the results we wanted when we arranged the controllable parameters with the PPO algorithm in order to minimize the amount of hydrogen wasted during hydrogen production. He received a low reward in 10000 and 100000 timestamps and did not reach the global optimum as depicted in Figure 5 and Figure 6. The necessity for more iterations and an extended execution period to achieve convergence was observed. Consequently, we came to the consensus that the performance of the PPO algorithm did not live up to our initial expectations.

On the other hand, both the DDPG and SAC algorithms have provided satisfactory results by adapting well to our problem and have achieved convergence. At this juncture, the SAC algorithm has taken a step ahead. It was observed that DDPG converged more rapidly.

In the system, the parameter that most significantly affects the wasted hydrogen during production and the amount of natural gas consumed is the C9 drum pressure. Consequently, we attempted to optimize this pressure value for better performance. The outcomes of the DRL algorithms that we implemented to optimize the value of the pressure-controlling sensor can be observed in Figure 5 and Figure 6. At this point, both the SAC and the DDPG algorithms have produced successful results and have converged.

These tests were conducted with different initial seeds and were evaluated at both 10000 and 100000 timestamps.

As evidenced by the graphs, the SAC algorithm has converged more swiftly and with a higher reward. The SAC algorithm is a powerful and flexible RL algorithm that can be used to solve a wide range of continuous control problems. SAC algorithm findings exhibited steadiness across all varied tests carried out.

Consequently, we selected SAC as our primary model.

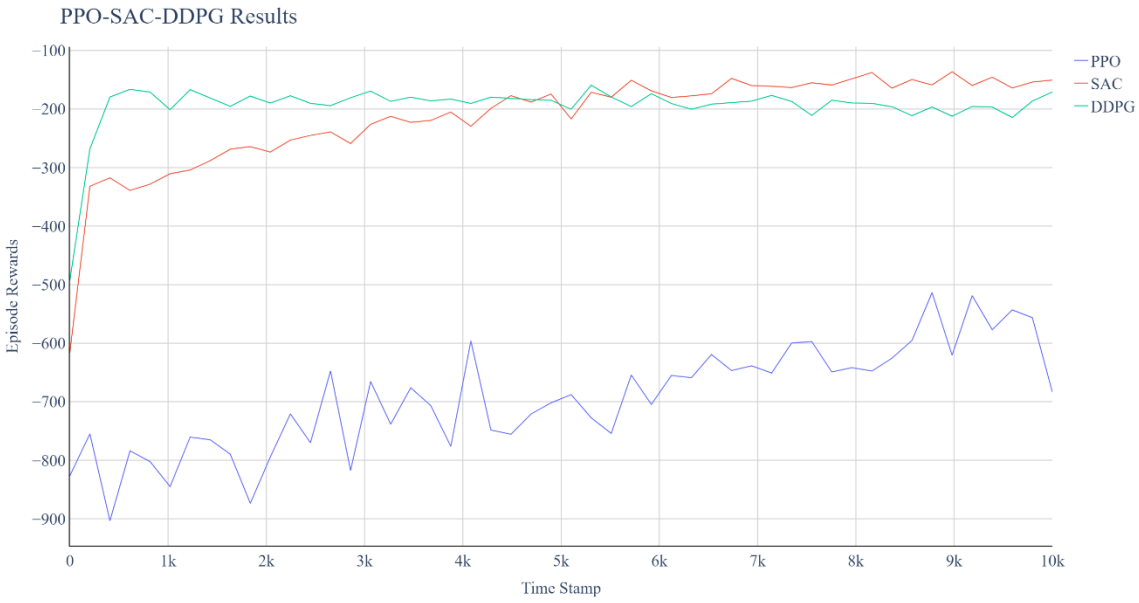

Figure 5. PPO, SAC, DDPG episode rewards for 10000.

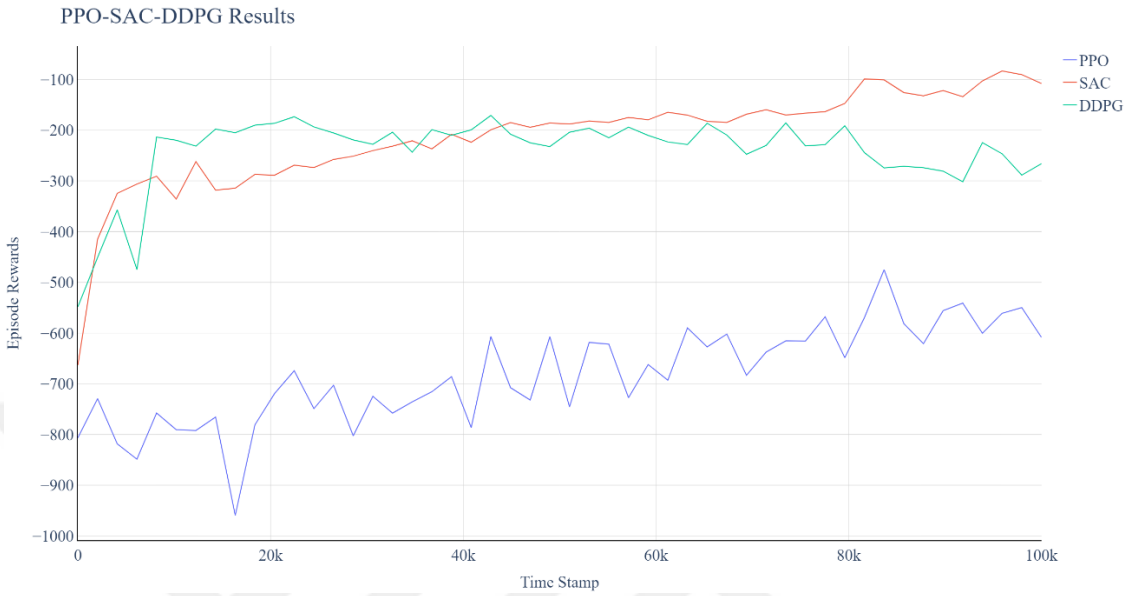

Figure 6. PPO, SAC, DDPG episode rewards for 100000.

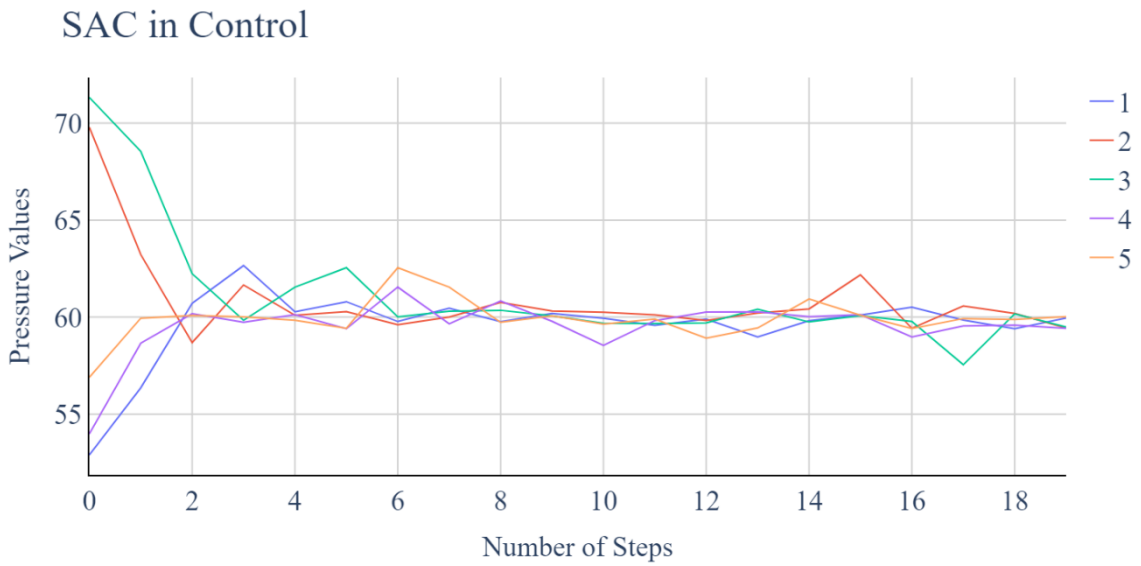

Figure 7. When the SAC agent is in the environment to optimize pressure value.

As shown in Figure 7, when we run our agent in the initial phase of the simulation environment we developed, it adjusts the C9 drum pressure sensor value to the desired state. This scenario has been tested with many different initial points, and every time it has been observed to converge to the desired point. In the initial phase, as mentioned, each

step has a resolution of 5 minutes. As seen in Figure 7, the five experiments we ran reached the desired region in 2 steps, which is equivalent to 10 minutes. Subsequently, it stabilizes within that desired region which is 60 bar. In our DRL model, the agent remained active for 100 episodes at a time, transferring its learnings to the subsequent episode. Upon training completion, the model achieved an average reward value of -127 over a run of 100 episodes. We applied the reward mechanism utilized in the DRL model by segmenting the C9 drum pressure values into batches of 100 episodes. As a result of this process, we obtained an average reward value of -138 from the real system. This indicates an improvement of approximately 9%. The critical point to emphasize is the success of our simulated environment. To arrive at a more accurate result, we need to deduct the error of our simulation from this 9% improvement. Our simulation error rate is seen to be approximately 5% for C9 drum pressure. See Target 11 from Table 5. Hence, we can observe an improvement of about 4%. According to field engineers, this improvement in pressure value leads to approximately a 4% increase in energy efficiency. The 4% improvement system in the hydrogen production unit, which consumes a high amount of natural gas, significantly enhances its performance and results in substantial savings. The detailed efficiency calculations will be calculated by field engineers along with the live field tests.

The most important advantage of this framework is its ability to optimize the system in a completely data-driven manner, without requiring any chemical formulation or understanding of the system's operational principles. This framework can be expanded based on the requests and guidance of field engineers and can be adapted to handle more complex relationships and control a greater number of parameters. Observing the results of the application in a live system will take time. Initially, when we shared the results with field experts, we received positive feedback. We questioned whether the designed framework would be utilized and if it could be integrated into their existing systems. They mentioned that this system provides significant findings, and they would like to use it in daily operations. In the initial phase, they wanted to use this simulation environment when making decisions, and in subsequent periods, they mentioned the possibility of connecting these to the APC. Also, as the training was done with the data of the unit they manage, it will be more beneficial than what external software offers. At this stage, we are preparing a more detailed test scenario in which intervene in the live system.

CHAPTER 4

CONCLUSION

In summary, this thesis study proposes a novel approach for optimizing hydrogen production in an energy manufacturing organization by utilizing a combination of a digital twin simulation environment and DRL. The overarching objective of the methodology is to enhance productivity and curtail the costs associated with hydrogen production by deploying advanced ML approaches.

This thesis study consists of two main phases: (1) Developing a data-driven simulation environment, (2) Optimizing hydrogen production by leveraging state-of-the-art DRL approaches. The first phase of this study involves creating a data-driven digital twin framework by utilizing advanced ML algorithms and approaches, which can be used to simulate the real process values and determine the best production operation states. The digital twin simulation model is designed for providing a highly interactive, generalizable, and scalable. This allows us to leverage this model in improving overall operational asset efficiency, reducing hydrogen and natural gas consumption by modeling the impact of various manipulated operating parameters and enabling real-time adjustments. Additionally, the proposed thesis can also be leveraged to train new field engineers for increasing their adoption in hydrogen production.

The second phase of this thesis study involves leveraging the developed digital twin simulation environment as the basis for a DRL application in optimizing hydrogen production. The developed ML models are trained using a real-world dataset extracted from the energy manufacturing organization for developing a DRL agent that can autonomously optimize the hydrogen production process by changing the manipulated process variables. Tn this phase, the PPO, SAC and DDPG algorithms were tested. Experiments were conducted and the SAC algorithm outperformed others. With our established framework, a data-driven approach is utilized in the hydrogen production unit, leading to a significantly reduced reliance on heavily formulaic modeling. As a result, a more generalized structure has been created, capable of responding much faster to sudden changes in the process due to its data-driven nature. The experiment results reveal that the proposed approach yields encouraging results by improving the overall performance of the hydrogen production up to %4. This improvement can be further enhanced with the future studies mentioned in the future work.

This study also comprises some limitations and important improvement opportunities to be tackled in future studies. Firstly, various ML algorithms were developed for each target process variable as a baseline model that utilizes the default hyperparameters and only the best performing baseline model was selected for tuning hyperparameters in improving model performance. However, when each ML algorithm is fine-tuned, they might also

provide better results. In this regard, with greater computational power, a more optimal model could potentially be identified.

The second limitation of this study is about the performance of the developed DRL agent. The DRL agent is directly related to the success of the DDSE that is developed in the context of this thesis. However, the performance of that model is highly tied to the success of DRL agent. The small errors or inaccuracies that may be provided by the DDSE can directly influence the agent's decisions, potentially leading it to make incorrect actions. To this end, it's crucial to regularly monitor and track the model metrics.

In future work, we are planning to improve the proposed framework by generalizing the application domain. The DDSE algorithm is mainly based on regression algorithms. However, the dataset can be divided into different production states by utilizing the state-of-the-art classification algorithms to ensemble regression algorithms for each production state in generalizing the DDSE models. Moreover, instead of predicting the observed variables' precise values, the DDSE models can be developed to predict whether the target values decline or increase within predefined process value ranges. This approach could mitigate the complexity of the problem and facilitate more accurate approximations for the DRL agent.

We are also planning to integrate the Human-in-the-loop DRL to this thesis study for incorporating expert feedback or other forms of human interaction to guide the DRL agent, especially in scenarios where the agent's actions might be uncertain or risky. This allows us to test the performance of the developed DRL agent and also improve the performance of the DRL agent continuously. Another important future work is about robustness testing which introduces more varied and challenging scenarios in the simulated environment to ensure the DRL model is robust to be applied in a wide range of operational challenges.

The Machine Learning Operations System (MLOps) can also be designed and established to monitor the performance indicators including RMSE and RRSE of the models in real-time. Moreover, some error thresholds can be defined to trigger retraining of the models with new datasets.

Furthermore, our goal is to extend the framework we have established to other production units in order to enhance efficiency across the entire production line. During the process of this dissemination, new approaches such as transfer learning can be employed.

Lastly, the performance DRL agent is highly affected by the reward function [58]. In this thesis study, the reward function is determined by the collaboration of the stakeholders including the domain experts and data scientists. However, we are also planning to expand our efforts in reward engineering by adapting it to various production facilities.

REFERENCES

[1] "Manufacturing, value added (% of GDP) | Data." Accessed: Aug. 28, 2023. [Online]. Available: https://data.worldbank.org/indicator/NV.IND.MANF.ZS

[2] K. Kaygusuz, "Energy and environmental issues relating to greenhouse gas emissions for sustainable development in Turkey," *Renewable and Sustainable Energy Reviews*, vol. 13, no. 1. 2009. doi: 10.1016/j.rser.2007.07.009.

[3] D. Wetherill, "Broken links Why analytics investments have yet to pay off," *ZS Associates*, 2016.

[4] A. Senthil Kumar and E. Iyer, "An industrial iot in engineering and manufacturing industries - Benefits and challenges," *International Journal of Mechanical and Production Engineering Research and Development*, vol. 9, no. 2, 2019, doi: 10.24247/ijmperdapr201914.

[5] A. Belhadi, K. Zkik, A. Cherrafi, S. M. Yusof, and S. El fezazi, "Understanding Big Data Analytics for Manufacturing Processes: Insights from Literature Review and Multiple Case Studies," *Comput Ind Eng*, vol. 137, 2019, doi: 10.1016/j.cie.2019.106099.

[6] M. Khorasani *et al.*, "A review of Industry 4.0 and additive manufacturing synergy," *Rapid Prototyping Journal*, vol. 28, no. 8. Emerald Group Holdings Ltd., pp. 1462–1475, Aug. 02, 2022. doi: 10.1108/RPJ-08-2021-0194.

[7] M. Yu, M. Zhu, G. Chen, J. Li, and Z. Zhou, "A cyber-physical architecture for industry 4.0-based power equipments detection system," in *CMD 2016 - International Conference on Condition Monitoring and Diagnosis*, 2016. doi: 10.1109/CMD.2016.7757942.

[8] N. Stojanovic and D. Milenovic, "Data-driven Digital Twin approach for process optimization: An industry use case," in *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, 2019. doi: 10.1109/BigData.2018.8622412.

[9] A. Tulsyan, S. R. Khare, B. Huang, R. B. Gopaluni, and J. F. Forbes, "Bayesian identification of non-linear state-space models: Part I-Input design," in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2013. doi: 10.3182/20131218-3-IN-2045.00105.

[10] M. Kano and M. Ogawa, “The state of the art in advanced chemical process control in Japan,” in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2009. doi: 10.3182/20090712-4-tr-2008.00005.

[11] M. A. Rosen and S. Koohi-Fayegh, “The prospects for hydrogen as an energy carrier: an overview of hydrogen energy and hydrogen energy systems,” *Energy, Ecology and Environment*, vol. 1, no. 1. 2016. doi: 10.1007/s40974-016-0005-z.

[12] L. Matijašević and M. Petrić, “Integration of hydrogen systems in petroleum refinery,” *Chem Biochem Eng Q*, vol. 30, no. 3, 2016, doi: 10.15255/CABEQ.2015.2337.

[13] A. Velazquez Abad and P. E. Dodds, “Production of Hydrogen,” *Encyclopedia of Sustainable Technologies*, pp. 293–304, Jul. 2017, doi: 10.1016/B978-0-12-409548-9.10117-4.

[14] M. Liu, S. Fang, H. Dong, and C. Xu, “Review of digital twin about concepts, technologies, and industrial applications,” *J Manuf Syst*, vol. 58, 2021, doi: 10.1016/j.jmsy.2020.06.017.

[15] K. Alexopoulos, N. Nikolakis, and G. Chryssolouris, “Digital twin-driven supervised machine learning for the development of artificial intelligence applications in manufacturing,” *Int J Comput Integr Manuf*, vol. 33, no. 5, 2020, doi: 10.1080/0951192X.2020.1747642.

[16] I. M. Cavalcante, E. M. Frazzon, F. A. Forcellini, and D. Ivanov, “A supervised machine learning approach to data-driven simulation of resilient supplier selection in digital manufacturing,” *Int J Inf Manage*, vol. 49, 2019, doi: 10.1016/j.ijinfomgt.2019.03.004.

[17] C. M. Bishop, “Bishop - Pattern Recognition And Machine Learning - Springer 2006,” *Antimicrob Agents Chemother*, vol. 58, no. 12, 2014.

[18] A. Ethem, *Introduction to Machine Learning Second Edition Adaptive Computation and Machine Learning*. 2015.

[19] B. Y. Goodfellow I., “Courville A-Deep learning-MIT (2016),” *Nature*, vol. 26, no. 7553, 2016.

[20] D. A. Freedman, *Statistical models: Theory and practice*. 2009. doi: 10.1017/CBO9780511815867.

[21] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning, Springer Texts*, vol. 102. 2013.

[22] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. doi: 10.1145/2939672.2939785.

[23] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, 2017.

[24] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 42, no. 1, 2000, doi: 10.1080/00401706.2000.10485983.

[25] R. Tibshirani, "Regression shrinkage and selection via the lasso: A retrospective," *J R Stat Soc Series B Stat Methodol*, vol. 73, no. 3, 2011, doi: 10.1111/j.1467-9868.2011.00771.x.

[26] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction. Cambridge: MIT Press.," *MA: MIT Press.[Google Scholar]*, 1998.

[27] V. Mnih, D. Silver, and M. Riedmiller, "Playing Atari with Deep Q Learning," *Nips*, 2013.

[28] M. Sugiyama, *Statistical reinforcement learning: Modern machine learning approaches*. 2015. doi: 10.1201/b18188.

[29] M. Lapan, "Deep Reinforcement Learning Hands-On. Apply modern RL methods, with deep Q-networks, value iteration, policy gradients, TRPO, AlphaGo zero and more," *Packt Publishing Ltd.* 2018.

[30] M. Kong and J. So, "Empirical Analysis of Automated Stock Trading Using Deep Reinforcement Learning," *Applied Sciences (Switzerland)*, vol. 13, no. 1, 2023, doi: 10.3390/app13010633.

[31] J. Zhang, Z. Zhang, S. Han, and S. Lü, "Proximal policy optimization via enhanced exploration efficiency," *Inf Sci (N Y)*, vol. 609, 2022, doi: 10.1016/j.ins.2022.07.111.

[32] W. Xue, H. Wu, H. Ye, and S. Shao, "An Improved Proximal Policy Optimization Method for Low-Level Control of a Quadrotor," *Actuators*, vol. 11, no. 4, 2022, doi: 10.3390/act11040105.

[33] A. Peng, S. L. Ang, and C. Y. Lim, "Automated Cryptocurrency Trading Bot Implementing DRL," *Pertanika J Sci Technol*, vol. 30, no. 4, 2022, doi: 10.47836/pjst.30.4.22.

[34] M. Grzelczak and P. Duch, “Deep reinforcement learning algorithms for path planning domain in grid-like environment,” *Applied Sciences (Switzerland)*, vol. 11, no. 23, 2021, doi: 10.3390/app112311335.

[35] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” in *35th International Conference on Machine Learning, ICML 2018*, 2018.

[36] C. C. Wong, S. Y. Chien, H. M. Feng, and H. Aoyama, “Motion planning for dual-arm robot based on soft actor-critic,” *IEEE Access*, vol. 9, 2021, doi: 10.1109/ACCESS.2021.3056903.

[37] T. P. Lillicrap *et al.*, “Continuous control with deep reinforcement learning,” in *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.

[38] J. Morgan, M. Halton, Y. Qiao, and J. G. Breslin, “Industry 4.0 smart reconfigurable manufacturing machines,” *Journal of Manufacturing Systems*, vol. 59. 2021. doi: 10.1016/j.jmsy.2021.03.001.

[39] P. K. R. Maddikunta *et al.*, “Industry 5.0: A survey on enabling technologies and potential applications,” *Journal of Industrial Information Integration*, vol. 26. 2022. doi: 10.1016/j.jii.2021.100257.

[40] “Rolls-Royce turns to digital twins to improve jet engine efficiency | CIO.” Accessed: Nov. 13, 2022. [Online]. Available: https://www.cio.com/article/188765/rolls-royce-turns-to-digital-twins-to-improve-jet-engine-efficiency.html

[41] D. P. Xenos *et al.*, “Optimization of a network of compressors in parallel: Real Time Optimization (RTO) of compressors in chemical plants - An industrial case study,” *Appl Energy*, vol. 144, 2015, doi: 10.1016/j.apenergy.2015.01.010.

[42] B. Marr and M. Ward, “Artificial intelligence in practice : how 50 successful companies used artificial intelligence to solve problems,” *Business & Economics*, 2019.

[43] X. Wang, Y. Wang, F. Tao, and A. Liu, “New Paradigm of Data-Driven Smart Customisation through Digital Twin,” *J Manuf Syst*, vol. 58, 2021, doi: 10.1016/j.jmsy.2020.07.023.

[44] K. Alexopoulos, N. Nikolakis, and G. Chryssolouris, “Digital twin-driven supervised machine learning for the development of artificial intelligence applications in manufacturing,” *Int J Comput Integr Manuf*, vol. 33, no. 5, 2020, doi: 10.1080/0951192X.2020.1747642.

[45] T. R. Wanasinghe, R. G. Gosine, L. A. James, G. K. I. Mann, O. De Silva, and P. J. Warrian, “The Internet of Things in the Oil and Gas Industry: A Systematic Review,” *IEEE Internet Things J*, vol. 7, no. 9, 2020, doi: 10.1109/JIOT.2020.2995617.

[46] B. T. Kumaravel, R. Bhattacharyya, J. Siegel, S. E. Sarma, and N. Arunachalam, “Development of an Internet of Things enabled manufacturing system for tool wear characterization,” in *2017 IEEE 3rd International Symposium in Robotics and Manufacturing Automation, ROMA 2017*, 2017. doi: 10.1109/ROMA.2017.8231733.

[47] “OpenAI.” Accessed: Jul. 19, 2023. [Online]. Available: https://openai.com/

[48] “Google DeepMind.” Accessed: Jul. 19, 2023. [Online]. Available: https://www.deepmind.com/

[49] H. Nguyen and H. La, “Review of Deep Reinforcement Learning for Robot Manipulation,” in *Proceedings - 3rd IEEE International Conference on Robotic Computing, IRC 2019*, 2019. doi: 10.1109/IRC.2019.00120.

[50] Q. Xing, Z. Chen, T. Zhang, X. Li, and K. H. Sun, “Real-time optimal scheduling for active distribution networks: A graph reinforcement learning method,” *International Journal of Electrical Power and Energy Systems*, vol. 145, 2023, doi: 10.1016/j.ijepes.2022.108637.

[51] T. Rupprecht and Y. Wang, “A survey for deep reinforcement learning in markovian cyber–physical systems: Common problems and solutions,” *Neural Networks*, vol. 153. 2022. doi: 10.1016/j.neunet.2022.05.013.

[52] J. Leng *et al.*, “A loosely-coupled deep reinforcement learning approach for order acceptance decision of mass-individualized printed circuit board manufacturing in industry 4.0,” *J Clean Prod*, vol. 280, 2021, doi: 10.1016/j.jclepro.2020.124405.

[53] R. Lu, Y. C. Li, Y. Li, J. Jiang, and Y. Ding, “Multi-agent deep reinforcement learning based demand response for discrete manufacturing systems energy management,” *Appl Energy*, vol. 276, 2020, doi: 10.1016/j.apenergy.2020.115473.

[54] J. Lee *et al.*, “Machine learning-based energy optimization for on-site SMR hydrogen production,” *Energy Convers Manag*, vol. 244, 2021, doi: 10.1016/j.enconman.2021.114438.

[55] H. Xu, L. Zhang, P. Li, and F. Zhu, “Outlier detection algorithm based on k-nearest neighbors-local outlier factor,” *J Algorithm Comput Technol*, vol. 16, 2022, doi: 10.1177/17483026221078111.

[56] D. Feng, F. Chen, and W. Xu, “Efficient leave-one-out strategy for supervised feature selection,” *Tsinghua Sci Technol*, vol. 18, no. 6, 2013, doi: 10.1109/tst.2013.6678908.

[57] R. Shwartz-Ziv and A. Armon, “Tabular data: Deep learning is not all you need,” *Information Fusion*, vol. 81, 2022, doi: 10.1016/j.inffus.2021.11.011.

[58] D. Dewey, “Reinforcement Learning and the Reward Engineering Principle,” in *2014 AAAI Spring Symposium Series*, 2014.

APPENDICES

APPENDIX-A RESULTS OF FIVE FOLD CROSS VALIDATION WITHOUT RANDOM SPLITTING

Table 8. Result table considering the time axis.

Models		**Linear**		**Lasso**		**Ridge**		**XGBoost**		**LightGBM**		**FFNN**	
		RMSE	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**	**RMSE**	**RMSE/STD**
Target 1	Train	0.147	0.521	0.262	0.927	0.147	0.521	**0.086**	**0.304**	0.112	0.398	0.230	0.814
	Test	0.210	0.741	0.365	1.289	0.210	0.741	**0.411**	**1.452**	0.170	0.600	0.238	0.840
Target 2	Train	0.051	0.118	0.305	0.698	0.051	0.118	0.029	0.067	**0.037**	**0.085**	0.390	0.892
	Test	0.041	0.093	0.425	0.972	0.041	0.093	0.086	0.197	**0.041**	**0.093**	0.414	0.945
Target 3	Train	578.34	0.051	580.91	0.051	578.34	0.051	401.80	0.035	**475.85**	**0.042**	813.16	0.072
	Test	753.79	0.066	680.91	0.060	753.79	0.066	1313.23	0.116	**765.19**	**0.067**	2384.77	0.211
Target 4	Train	175.27	0.026	179.22	0.027	175.27	0.026	**101.88**	**0.015**	132.76	0.020	249.00	0.038
	Test	182.29	0.028	153.52	0.023	182.12	0.027	**371.61**	**0.057**	267.93	0.041	853.15	0.131

Table 8 Cont.

Target 5	Train	0.207	0.019	**0.766**	**0.071**	0.207	0.019	0.157	0.014	0.183	0.017	0.324	0.054
	Test	0.249	0.023	**1.781**	**0.167**	0.249	0.023	0.600	0.056	0.470	0.044	0.693	0.030
Target 6	Train	522.27	0.057	531.99	0.058	522.27	0.057	325.50	0.035	409.35	0.044	605.75	0.066
	Test	762.57	0.083	852.78	0.093	762.57	0.083	1796.91	0.197	852.47	0.093	1494.82	0.164
Target 7	Train	1.938	0.099	2.084	0.107	1.938	0.099	**1.172**	**0.060**	1.463	0.075	1.717	0.088
	Test	3.658	0.187	2.193	0.112	3.658	0.187	**9.220**	**0.473**	3.913	0.200	4.716	0.242
Target 8	Train	2.059	0.105	2.159	0.110	2.059	0.105	1.239	0.063	**1.541**	**0.078**	1.850	0.094
	Test	3.825	0.195	2.416	0.123	3.825	0.195	8.932	0.457	**4.293**	**0.219**	4.215	0.215
Target 9	Train	474.39	0.071	475.38	0.071	474.39	0.071	**319.49**	**0.047**	382.83	0.057	816.12	0.122
	Test	704.93	0.105	673.75	0.100	704.93	0.105	**1223.29**	**0.183**	729.56	0.109	942.93	0.141
Target 10	Train	278.17	0.063	279.53	0.063	278.17	0.063	179.82	0.040	**220.09**	**0.050**	253.68	0.057
	Test	374.11	0.085	354.22	0.080	374.11	0.085	812.70	0.184	**446.95**	**0.101**	1484.18	0.337
Target 11	Train	4.021	0.773	**4.069**	**0.782**	4.021	0.773	2.773	0.533	3.305	0.635	3.603	0.692
	Test	4.464	0.858	**4.565**	**0.877**	4.464	0.858	6.270	1.205	4.388	0.843	4.473	0.860
Target 12	Train	0.243	0.340	0.704	0.982	0.243	0.340	0.183	0.256	**0.216**	**0.301**	0.232	0.324
	Test	0.205	0.286	1.792	2.499	0.205	0.286	0.292	0.408	**0.209**	**0.292**	0.337	0.470

Table 8 Cont.

Target 13	Train	0.203	0.283	0.614	0.856	0.203	0.283	0.154	0.214	**0.186**	**0.259**	0.207	0.288
	Test	0.196	0.273	1.531	2.135	0.196	0.273	0.261	0.363	**0.177**	**0.246**	0.304	0.423
Target 14	Train	651.03	0.686	651.27	0.686	651.03	0.686	463.77	0.488	**547.96**	**0.577**	582.58	0.614
	Test	1045.12	1.101	1048.44	1.105	1045.12	1.101	1228.23	1.294	**1052.60**	**1.109**	1162.12	1.225
Target 15	Train	1.987	0.100	2.131	0.108	1.987	0.100	**1.235**	**0.062**	1.530	0.077	1.771	0.090
	Test	3.847	0.195	2.290	0.116	3.847	0.195	**9.238**	**0.469**	4.900	0.248	4.151	0.210
Target 16	Train	2.027	0.104	2.154	0.110	2.027	0.104	**1.248**	**0.064**	1.542	0.079	1.893	0.097
	Test	3.657	0.188	2.218	0.114	3.657	0.188	**8.566**	**0.440**	4.509	0.231	4.224	0.217

APPENDIX-B SELECTED FEATURES FOR EACH TARGET AFTER THE FEATURE SELECTION

Table 9. Final selected features for each target.

Features	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12	Y13	Y14	Y15	Y16
X1				*	*		*	*				*				
X2			*		*			*			*		*			
X3						*		*							*	
X4																
X5				*	*										*	
X6																
X7																
X8					*			*								
X9			*								*			*		
X10		*														
X11		*	*	*	*		*	*			*	*	*			*
X12			*				*								*	*
X13			*	*	*		*									
X14								*				*	*			
X15				*	*									*		
X16				*			*									
X17								*			*			*		
X18		*	*		*		*	*								
X1_diff											*					
X2_diff								*								
X3_diff	*		*		*		*	*	*		*	*	*	*	*	*
X4_diff		*	*				*	*			*	*	*	*		*
X5_diff			*	*	*					*		*				
X6_diff																
X7_diff		*	*				*	*			*			*		
X8_diff			*				*	*			*		*	*	*	*
X9_diff			*									*		*		
X10_diff			*		*			*			*			*		
X11_diff		*	*		*		*	*			*		*	*		*
X12_diff		*	*		*		*	*			*		*	*		
X13_diff											*					
X14_diff			*				*	*			*	*		*		
X15_diff		*	*	*	*		*	*			*			*		
X16_diff			*	*	*		*	*			*			*		

Table 9 Cont.

X17_diff							*				*			*		
X18_diff					*			*			*			*		
X1^2																
X1 X2					*											
X1 X3			*	*	*							*				
X1 X4	*		*												*	
X1 X5		*			*											*
X1 X6												*				
X1 X7			*	*	*		*	*			*	*				
X1 X8					*			*				*		*		
X1 X9		*						*				*	*			
X1 X10					*							*				
X1 X11	*				*	*	*	*	*	*		*	*	*	*	*
X1 X12		*	*	*	*		*	*			*		*			
X1 X13	*	*	*	*	*		*	*			*	*	*	*	*	*
X1 X14												*				
X1 X15																
X1 X16			*		*											
X1 X17												*				
X1 X18			*		*							*				
X2^2																
X2 X3												*				
X2 X4				*												
X2 X5																
X2 X6	*				*				*				*			
X2 X7					*		*						*		*	*
X2 X8	*		*		*		*	*			*	*	*		*	*
X2 X9					*											
X2 X10					*							*				
X2 X11					*		*	*			*	*				
X2 X12		*	*	*	*		*	*				*				
X2 X13											*	*	*	*		
X2 X14			*											*		
X2 X15																
X2 X16			*	*			*					*	*	*		
X2 X17								*				*				
X2 X18			*	*				*								
X3^2																
X3 X4			*													
X3 X5				*												

Table 9 Cont.

X3 X6					*							*				
X3 X7			*					*				*				
X3 X8		*	*					*			*					
X3 X9			*													
X3 X10				*	*											
X3 X11																*
X3 X12			*				*	*								
X3 X13			*													
X3 X14			*			*		*	*	*					*	
X3 X15																
X3 X16		*	*	*	*			*					*			
X3 X17				*	*											
X3 X18				*			*					*				
X4^2																
X4 X5				*								*				
X4 X6					*											
X4 X7		*	*				*					*	*			
X4 X8		*						*				*				*
X4 X9			*													
X4 X10																
X4 X11							*									
X4 X12			*					*			*					
X4 X13			*				*	*					*			*
X4 X14			*				*									
X4 X15																
X4 X16			*	*												
X4 X17				*	*							*				
X4 X18								*						*		
X5^2												*				
X5 X6			*	*	*		*	*			*			*		
X5 X7			*	*			*		*	*		*	*		*	*
X5 X8			*	*	*		*						*			
X5 X9		*														
X5 X10	*	*	*	*	*		*	*				*	*			*
X5 X11		*														
X5 X12		*	*	*	*		*									*
X5 X13			*	*	*			*		*						
X5 X14																
X5 X15																
X5 X16			*		*			*				*				

Table 9 Cont.

X5 X17			*						*			*				*
X5 X18				*	*											
X6^2		*	*	*			*	*					*		*	*
X6 X7					*							*				
X6 X8		*	*	*	*		*	*			*	*		*		
X6 X9				*												
X6 X10			*	*	*		*	*				*	*			
X6 X11								*								*
X6 X12			*													
X6 X13			*	*	*		*	*			*			*		
X6 X14		*											*			
X6 X15							*									
X6 X16			*	*	*		*	*				*				
X6 X17				*				*								
X6 X18	*		*	*	*		*	*	*	*	*	*	*		*	*
X7^2																
X7 X8	*	*	*	*	*	*	*		*	*		*	*		*	*
X7 X9					*											
X7 X10				*	*		*	*				*	*			*
X7 X11		*											*			
X7 X12		*	*		*		*									
X7 X13			*	*	*			*			*	*		*		
X7 X14	*	*			*		*					*				
X7 X15																
X7 X16				*	*		*	*				*				
X7 X17			*		*			*				*				
X7 X18		*	*	*	*		*	*			*	*	*	*	*	*
X8^2																
X8 X9													*			
X8 X10			*		*			*				*				
X8 X11					*											
X8 X12														*		
X8 X13	*	*	*	*	*	*	*	*	*	*	*	*	*		*	*
X8 X14																
X8 X15						*					*		*	*		
X8 X16	*			*			*	*				*	*		*	*
X8 X17			*	*	*		*	*				*	*	*		
X8 X18		*	*	*	*		*	*				*	*		*	*
X9^2																
X9 X10			*													

Table 9 Cont.

X9 X11																
X9 X12			*													
X9 X13			*				*									
X9 X14		*	*	*		*		*		*	*	*	*			*
X9 X15	*	*									*					
X9 X16		*		*			*									
X9 X17				*										*		
X9 X18		*			*											
X10^2																
X10 X11	*		*		*		*						*		*	*
X10 X12		*		*	*			*				*		*		
X10 X13	*	*	*	*	*	*	*	*	*	*	*	*	*		*	*
X10 X14					*			*								
X10 X15		*	*													
X10 X16	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
X10 X17			*		*							*		*		
X10 X18	*	*		*	*	*	*	*	*	*	*	*	*	*	*	*
X11^2																
X11 X12		*	*		*		*	*				*				
X11 X13	*	*	*	*		*	*	*	*	*		*	*		*	*
X11 X14			*									*				
X11 X15		*	*	*	*		*	*				*	*	*		*
X11 X16			*	*												
X11 X17			*	*	*			*						*	*	*
X11 X18		*	*	*	*		*	*			*	*	*	*		*
X12^2																
X12 X13		*	*	*	*		*	*			*	*		*		
X12 X14			*													
X12 X15							*	*					*			
X12 X16		*	*	*	*											
X12 X17			*	*												*
X12 X18			*	*	*			*						*		
X13^2																
X13 X14				*	*											
X13 X15																
X13 X16		*	*	*	*		*	*				*				
X13 X17		*	*	*	*		*	*								
X13 X18			*									*				
X14^2																
X14 X15		*	*					*				*	*			

Table 9 Cont.

X14 X16			*	*	*											
X14 X17																
X14 X18		*	*		*			*					*			
X15^2																
X15 X16				*												
X15 X17											*					
X15 X18				*				*								
X16^2																
X16 X17				*	*			*				*		*		
X16 X18			*		*							*				
X17^2														*		
X17 X18							*	*	*			*			*	*
X18^2																
X1/X2		*	*		*		*	*				*		*	*	*
X1/X3																
X1/X4			*	*												
X1/X5				*	*			*						*		
X1/X6	*	*	*	*	*		*	*	*	*		*	*		*	*
X1/X7		*			*		*	*			*	*				
X1/X8		*	*		*		*	*				*		*		
X1/X9			*		*							*			*	
X1/X10		*	*	*	*		*					*	*			*
X1/X11	*			*		*	*	*	*	*	*		*	*	*	*
X1/X12							*	*								
X1/X13		*	*		*		*	*			*		*	*		
X1/X14		*	*													
X1/X15		*			*						*		*	*		
X1/X17			*	*			*	*				*				
X1/X18		*	*	*	*		*	*				*				
X2/X3				*									*			
X2/X4			*	*					*	*						
X2/X5												*			*	
X2/X6			*	*	*		*	*				*	*			*
X2/X7		*			*	*						*	*			
X2/X8				*	*		*									
X2/X9		*						*							*	
X2/X10		*														
X2/X11		*			*		*	*			*	*		*		
X2/X12			*		*			*							*	*
X2/X13		*	*	*	*			*								

Table 9 Cont.

X2/X14											*					
X2/X15							*									
X2/X17												*			*	
X2/X18			*		*		*	*					*		*	*
X3/X4							*	*				*				
X3/X5				*								*				
X3/X6			*	*	*			*				*				
X3/X7					*											
X3/X8				*	*											
X3/X9											*		*	*		
X3/X10	*					*	*									
X3/X11			*													
X3/X12				*	*			*								
X3/X13			*	*												
X3/X14																
X3/X15																
X3/X17																
X3/X18				*												
X4/X5	*	*	*	*				*								
X4/X6		*	*	*	*		*	*				*				
X4/X7	*			*			*						*		*	
X4/X8		*			*											
X4/X9	*	*		*		*		*							*	*
X4/X10			*	*	*		*							*		
X4/X11			*	*			*									
X4/X12	*	*		*	*	*	*	*		*					*	*
X4/X13				*												
X4/X14	*					*		*	*		*			*		*
X4/X15			*													
X4/X17																
X4/X18			*	*					*	*						*
X5/X6			*	*	*			*				*		*		
X5/X7		*		*	*			*				*				
X5/X8		*	*	*	*			*				*	*			*
X5/X9							*									
X5/X10			*									*				
X5/X11		*		*	*		*									
X5/X12												*				
X5/X13			*	*	*			*				*				
X5/X14																

Table 9 Cont.

X5/X15																
X5/X17					*		*				*					*
X5/X18	*	*	*		*		*	*	*			*				
X6/X7		*		*	*		*	*			*	*	*		*	
X6/X8	*	*	*		*	*	*	*	*	*	*	*	*		*	*
X6/X9			*	*									*			
X6/X10			*													
X6/X11							*	*								
X6/X12			*	*	*		*	*								
X6/X13					*		*					*		*		
X6/X14																
X6/X15					*										*	
X6/X17			*													
X6/X18		*			*						*	*		*		
X7/X8			*	*	*		*	*			*	*		*		
X7/X9	*						*					*	*		*	*
X7/X10		*			*		*	*			*	*				
X7/X11					*		*					*				
X7/X12		*		*	*		*	*			*	*				
X7/X13		*	*	*	*		*	*			*	*	*	*	*	*
X7/X14			*				*	*			*	*	*			*
X7/X15																
X7/X17				*		*	*		*			*	*			
X7/X18		*	*	*	*		*	*			*	*	*	*		
X8/X9	*	*	*		*			*			*		*		*	*
X8/X10					*			*				*				
X8/X11	*		*	*	*		*						*			*
X8/X12	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
X8/X13			*		*			*				*		*		
X8/X14			*											*		
X8/X15												*				
X8/X17			*	*												
X8/X18			*									*				
X9/X10																
X9/X11				*							*					
X9/X12		*	*		*		*	*			*				*	*
X9/X13			*	*			*									
X9/X14												*				
X9/X15																
X9/X17											*			*		

Table 9 Cont.

X9/X18			*													
X10/X11		*	*				*							*		
X10/X12			*		*		*					*				
X10/X13	*	*		*	*		*				*	*			*	
X10/X14																
X10/X15												*				
X10/X17					*		*				*		*	*		
X10/X18					*											
X11/X12		*	*				*	*								
X11/X13		*	*	*	*		*	*				*	*			*
X11/X14			*	*											*	
X11/X15			*	*	*		*	*				*				
X11/X17			*				*	*			*			*		
X11/X18		*	*	*	*		*	*				*				*
X12/X13		*	*		*		*	*				*				
X12/X14			*													
X12/X15			*												*	*
X12/X17				*					*	*			*			
X12/X18			*	*	*			*				*				
X13/X14			*	*			*				*		*			
X13/X15				*												*
X13/X17				*	*											
X13/X18		*	*	*	*		*	*			*	*		*		
X14/X15	*												*			
X14/X17								*								
X14/X18			*				*	*								
X15/X17				*							*		*	*		
X15/X18				*				*								
X16/X17			*													
X16/X18		*	*	*	*							*				
X17/X18		*	*	*	*	*	*	*	*	*	*			*	*	*