

الجمهورية الشعبية الديمقراطية الجزائرية  
République Algérienne Démocratique et Populaire  
وزارة التعليم العالي والبحث العلمي  
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

المدرسة العليا للإعلام الآلي - 08 ماي 1945 – بسيدي بلعباس  
Ecole Supérieure en Informatique  
-08 Mai 1945- Sidi Bel Abbès



## Mémoire de Fin d'étude

Pour l'obtention du diplôme d'ingénieur d'état

Filière : Informatique

Spécialité : Ingénierie des Systèmes Informatiques (ISI)

## Thème

---

**The Use of Cognitive Digital Twins on an IoT System  
for Edge Resilience and Anomaly Detection**

---

Présenté par :

- Mme. SMATI Meriem

Soutenu le : **04/07/2023**

Devant le jury composé de :

- |                         |              |
|-------------------------|--------------|
| - M. BENSENANE Hamdane  | Président    |
| - M. RAHMOUN Abdellatif | Encadrant    |
| - M. LAVAL Jannik       | Encadrant    |
| - M. NIANG Boubou-Thiam | Co-Encadrant |
| - M. KHALDI Miloud      | Examineur    |

Année Universitaire : 2022 / 2023

# Acknowledgments

*First and foremost, I begin by expressing my deepest gratitude and thanks to ALLAH for granting me the opportunities, guidance, good health and strength to accomplish this work and achieve this valuable success.*

*I would also like to extend my heartfelt appreciation to my most important people, the ones that gave me life: my parents, Abdelmadjid Smati and Samira Seddiki, for their endless love, encouragement, and support. Their belief in my abilities and their sacrifices have been the driving force behind my accomplishments. I am profoundly grateful for their guidance and presence in my life.*

*To my sisters, Wissem and Sara, and my brother Mohamed, I am indebted for their constant support and understanding. Their presence and encouragement have been a source of strength and motivation throughout my internship journey.*

*To my grandmothers, that have been cheering me up all along.*

*To my aunt Amel Hammouche, I am grateful for your presence and constant support throughout both favorable and challenging moments. Your continuous encouragement and guidance have been truly appreciated.*

*I would also like to express my heartfelt gratitude to my childhood friend, Ines, my friends Sarah, Abir and Ahmed, thank you for your contribution in this project,*

*Your assistance in both technical and emotional aspects, thank you for your friendship, laughter, and support, it has been a constant source of strength and joy.*

*Special gratitude is also extended to my supervisors, Abdellatif Rahmoun, Jannik*

*Laval and Boubou Thiam Niang, for their valuable guidance, expertise, and mentorship. Their support and dedication have been instrumental in shaping my work and expanding my knowledge.*

*To all the ESI SBA family, to all these individuals, I extend my sincerest thanks.*

“The Use of Cognitive Digital Twins on an IoT System for Edge Resilience and Anomaly Detection” Engineering Thesis



*Your belief in me, your unwavering support, and the love and encouragement you have shown have been crucial in my personal and professional growth. I am deeply grateful for the impact you have had on my life and for being part of this important chapter.*

# Dedication

*I dedicate this humble work to my dear parents, you are the individuals who instilled in me the values of life and insured I had everything I needed and showered me with love, affection and joy.*

*In the memory of my grand fathers, papi and jeddou Lkhloufi who taught us the importance of never giving up and persevering in our endeavors.*

*To my lovely grandmothers, mami and mamara whose unwavering efforts are aimed at our happiness.*

*To my brother and sisters, Wissem, Sara and Mohamed who have consistently provided their support.*

*To my aunt Tatina who accompanied me during my whole journey and contributed in this accomplishment.*

*May they find here all my affection and love.*

# Abstract

The concept of Digital Twins (DTs) has progressed to encompass cognitive abilities, resulting in the emergence and appearance of Cognitive Digital Twins (CDTs).

CDTs are virtual representations of tangible or physical systems that have been enhanced with cognitive capabilities to carry out independent activities and autonomous tasks. They consist of a collection of interconnected digital models that can handle various types of data and descriptive and simulation models. The idea of CDTs enhances the cognitive capabilities of DTs using semantic technologies, enabling them to become more intelligent, all-encompassing, and capable of providing a complete representation of complex systems throughout their entire life cycle.

In this Engineering degree report, the main aspects and appliance of Digital Twin (DT) in resilience and anomaly detection based Machine Learning and Deep Learning approaches have been presented in the form of a a state of the art and led to the presentation of an experimental work that consists of developing a Cognitive Super-Digital Twin (CSDT) which not only replicates the actions of a system but also generates perturbations and anomalies as a means to bolster the system's security and ensure its continuity. It can identify vulnerabilities and devise appropriate countermeasures. This proactive approach enables the system to adapt and fortify its security measures, mitigating potential risks and ensuring uninterrupted operation.

**Keywords:** Digital Twins; Cognitive Digital Twins ; Artificial Intelligence; Machine Learning; Deep Learning; Internet of Things; Resilience;

“The Use of Cognitive Digital Twins on an IoT System for Edge Resilience and Anomaly Detection” Engineering Thesis

# Résumé

Le concept des Jumeaux Numériques (DT, pour Digital Twins) a évolué pour inclure des capacités cognitives, conduisant à l'émergence des Jumeaux Numériques Cognitifs (CDT, pour Cognitive Digital Twins).

Les CDTs sont des représentations numériques de systèmes physiques augmentées de capacités cognitives pour exécuter des activités autonomes. Ils comprennent un ensemble de modèles numériques sémantiquement interconnectés qui permettent de lier et récupérer des données hétérogènes, ainsi que des modèles descriptifs et de simulation. Le concept de CDT améliore les capacités cognitives des DT grâce aux technologies sémantiques, ce qui les rend plus intelligents, complets et capables de représenter l'ensemble du cycle de vie des systèmes complexes.

Dans ce rapport de diplôme d'ingénieur, les principaux aspects et applications des DT dans la résilience et la détection d'anomalies basées sur des approches d'apprentissage automatique et d'apprentissage profond ont été présentés sous la forme d'un état de l'art. Cela a conduit à la présentation d'un travail expérimental consistant à développer un CDT qui, non seulement reproduit les actions d'un système, mais génère également des perturbations et des anomalies afin de renforcer la sécurité du système et garantir sa continuité. Il peut identifier les vulnérabilités et élaborer des contre-mesures appropriées. Cette approche proactive permet au système de s'adapter et de renforcer ses mesures de sécurité, atténuant les risques potentiels et assurant un fonctionnement ininterrompu.

**Mots Clés:** Jumeaux numériques ; Jumeaux numériques cognitifs ; Internet des objets ; Résilience ; IA ; Apprentissage automatique ; Apprentissage profond

“The Use of Cognitive Digital Twins on an IoT System for Edge Resilience and Anomaly Detection” Engineering Thesis

# ملخص

تطور مفهوم التوائم الرقمية (DTs) ليشمل القدرات المعرفية ، مما أدى إلى ظهور التوائم الرقمية المعرفية (CDTs).

CDTs هي تمثيلات رقمية للأنظمة المادية التي يتم تعزيزها بالقدرات المعرفية لتنفيذ الأنشطة المستقلة. وهي تتألف من مجموعة من النماذج الرقمية المترابطة لغويًا والمتعلقة بربط واسترجاع البيانات غير المتجانسة ، فضلاً عن النماذج الوصفية والمحاكاة. يعزز مفهوم CDT القدرات الإدراكية لـ DTs باستخدام التقنيات الدلالية ، مما يمكنهم من أن يكونوا أكثر ذكاءً وشموليةً ، ويوفر تمثيلاً كاملاً لدورة الحياة للأنظمة المعقدة.

في تقرير الدرجة الهندسية هذا ، تم تقديم الجوانب والأجهزة الرئيسية لـ Digital Twin (DT) في مناهج التعلم الآلي والتعلم العميق القائمة على المرونة والكشف عن الشذوذ في شكل حالة من الفن وأدت إلى تقديم عمل تجريبي التي تتكون من تطوير التوأم الرقمي الفائق المعرفي (CSDT) الذي لا يكرر فقط إجراءات النظام ولكنه أيضاً يولد الاضطرابات والشذوذ كوسيلة لتعزيز أمن النظام وضمان استمراريته. يمكنه تحديد نقاط الضعف واستنباط التدابير المضادة المناسبة. يمكن هذا النهج الاستباقي النظام من تكييف تدابير الأمانة وتعزيزها ، وتخفيف المخاطر المحتملة وضمان التشغيل دون انقطاع.

**الكلمات الرئيسية:** التوائم الرقمية. التوائم الرقمية المعرفية. انترنت الأشياء؛ الذكاء الاصطناعي؛ التعلم الآلي؛ تعلم عميق. الذكاء الاصطناعي؛ التعلم الآلي؛ تعلم عميق

# Contents

<b>I</b>	<b>General Introduction</b>	<b>1</b>
<b>1</b>	<b>General Introduction</b>	<b>2</b>
1	Context . . . . .	3
2	Problem Statement . . . . .	3
3	Objectives . . . . .	4
4	Thesis Organization . . . . .	4
<b>II</b>	<b>Background and Definitions</b>	<b>6</b>
<b>2</b>	<b>Background</b>	<b>7</b>
1	Internet of Things (IoT) . . . . .	9
1.1	Introduction . . . . .	9
1.2	Definition of IoT . . . . .	9
1.3	Historical Evolution of IoT . . . . .	10
1.4	Characteristics of IoT . . . . .	11
1.5	IoT Architectures . . . . .	12
1.5.1	The Three and Five Layered Architectures . . . . .	13
1.5.2	The Edge Fog Cloud . . . . .	15
2	Internet of Things Resilience . . . . .	16
2.1	Introduction . . . . .	16
2.2	Definition of Resilience . . . . .	17
2.3	IoT resilience . . . . .	17

3	Machine Learning . . . . .	18
3.1	Definition of Machine Learning . . . . .	18
3.2	Types of Machine Learning . . . . .	18
3.2.1	Supervised Learning. . . . .	19
3.2.2	Unsupervised Learning . . . . .	20
3.2.3	Reinforcement Learning. . . . .	21
3.3	Machine Learning Models . . . . .	21
3.3.1	Decision Trees . . . . .	21
3.3.2	Random Forests . . . . .	23
3.3.3	Naive Bayes . . . . .	23
3.3.4	K-Nearest Neighbors . . . . .	24
4	Deep Learning . . . . .	25
4.1	Definition of Deep Learning . . . . .	25
4.2	Deep Learning Models . . . . .	25
4.2.1	CNNs. . . . .	26
4.2.2	Multilayer Perceptrons . . . . .	27
4.2.3	Reccurent Neural Networks. . . . .	28
5	Machine Learning VS Deep Learning . . . . .	30
6	Digital Twins . . . . .	33
6.1	Digital Twins History . . . . .	33

### **III State of the Art 37**

#### **3 State of the Art 38**

1	Digital Twins Concepts . . . . .	40
1.1	Introduction . . . . .	40
1.2	Definition of Digital Twins . . . . .	40
1.2.1	Deducing a General Definition of Digital Twins . . . . .	41
1.3	Components of a Digital Twin . . . . .	43

1.4	Characteristics and Requirements of a Digital Twin . . . . .	46
1.4.1	Essential Characteristics (Requirements) . . . . .	46
1.4.2	Dynamic Characteristics . . . . .	47
1.4.3	Key Characteristics Highlighted in this Paper . . . . .	49
1.5	An Overview on the Predecessors of Digital Twins and Their Key Differences . . . . .	49
1.5.1	Digital Model / Digital Simulation Model . . . . .	50
1.5.2	Digital Shadow . . . . .	50
1.5.3	Digital Twin . . . . .	51
1.5.4	Digital Model VS Digital Shadow VS Digital Twin . . . . .	52
1.6	Different Types of Digital Twins . . . . .	52
1.6.1	Components Twins . . . . .	53
1.6.2	Asset Twins . . . . .	53
1.6.3	System Twins/Unit Twins . . . . .	53
1.6.4	Process Twins . . . . .	54
2	Digital Twins with Machine Learning (ML) and DL . . . . .	56
2.1	Introduction . . . . .	56
2.2	ML Appliance in Digital Twins . . . . .	56
2.2.1	Diagnostic and Predictive Analytics: . . . . .	56
2.2.2	Prescriptive Analytics: . . . . .	57
2.3	Selecting an Adapted Model for IoT tabular Data . . . . .	59
2.3.1	Design and development of RNN anomaly detection model for IoT networks . . . . .	61
2.3.1.1	Description . . . . .	61
2.3.2	Why do tree-based models still outperform deep learning on tabular data? . . . . .	68
3	The use of Digital Twins for Resilience and Prevention . . . . .	70
3.1	Digital twins as run-time predictive models for the resilience of cyber-physical systems: a conceptual framework . . . . .	70



3.2	Cognitive Digital Twins for Resilience in Production: A Conceptual Framework . . . . .	73
3.3	State of the Art in using Digital Twins for prevention . . . . .	74
4	Digital Twins architecture . . . . .	76
5	Framework CSDT . . . . .	78
5.1	Conclusion . . . . .	79
<b>IV</b>	<b>Contribution</b>	<b>80</b>
<b>4</b>	<b>Design and Implementation</b>	<b>81</b>
1	Introduction . . . . .	83
2	Use Case . . . . .	83
2.1	Description of the Use Case . . . . .	83
2.2	Description of the used Dataset . . . . .	84
3	Used Technologies and Hardware . . . . .	85
3.1	Raspberry Pi 3 Model B . . . . .	85
3.1.1	Features and Specifications. . . . .	86
3.1.2	Chosen OS . . . . .	87
3.2	GrovePi+ . . . . .	87
3.3	Grove Sensors . . . . .	87
3.3.1	Grove Temperature&Humidity Sensor (DHT11) . . . . .	88
3.3.2	Grove - Barometer (High-Accuracy). . . . .	89
3.3.3	Grove - Light Sensor . . . . .	89
3.3.4	Grove-VOC and eCO2 Gas Sensor(SGP30) . . . . .	90
3.4	RabbitMQ - MQTT . . . . .	91

3.4.1	RabbitMQ . . . . .	91
3.4.2	RabbitMQ MQTT . . . . .	91
3.4.3	Why RabbitMQ - MQTT and not another Message broker - protocol ? . . . . .	92
3.5	InfluxDB . . . . .	93
3.6	Poppy Ergo Jr . . . . .	93
3.7	Computer . . . . .	94
4	General Architecture . . . . .	94
4.1	Simplified Architecture of the System . . . . .	94
4.1.1	Description of the Simplified Architecture . . . . .	95
4.1.1.1	Physical Twin . . . . .	95
4.1.1.2	Communication Medium . . . . .	96
4.1.1.3	Digital Twin . . . . .	97
4.2	Detailed Architecture of the System . . . . .	97
4.2.1	Description of the Detailed Architecture . . . . .	99
4.2.1.1	Physical Twin . . . . .	99
4.2.1.2	Communication Medium . . . . .	99
4.2.1.3	Digital Twin . . . . .	100
4.3	Elaborated Architecture of the System . . . . .	101
4.3.1	Description of the Elaborated Architecture . . . . .	103
4.3.1.1	Physical Twin . . . . .	103
4.3.1.2	Communication Medium . . . . .	106
4.3.1.3	Digital Twin . . . . .	107
5	Implementation . . . . .	108
5.1	Class Diagram . . . . .	108
5.1.1	Description of the Class Diagram . . . . .	108
5.2	Creating a Dataset . . . . .	120

5.2.1	The External Dataset . . . . .	121
5.2.2	The Global Dataset . . . . .	125
5.3	Selecting an ML or DL Models . . . . .	126
5.3.1	The Physical Twin’s Model. . . . .	126
5.3.1.1	Decision Tree . . . . .	127
5.3.1.2	Random Forests . . . . .	128
5.3.1.3	KNN . . . . .	130
5.3.1.4	Naive Bayes . . . . .	131
5.3.1.5	MLP . . . . .	133
5.3.1.6	RNN . . . . .	135
5.3.1.7	The PT Model selection . . . . .	137
5.3.2	The Digital Twin’s Model . . . . .	139
5.3.2.1	Decision Tree . . . . .	141
5.3.2.2	Random Forest . . . . .	144
5.3.2.3	KNN . . . . .	144
5.3.2.4	The DT Model Selection . . . . .	147
5.4	Languages and Libraries . . . . .	148
<b>5</b>	<b>Demonstration</b>	<b>150</b>
1	The Physical Twin . . . . .	151
1.1	The first sub-system . . . . .	151
1.1.1	The first sub-system edge. . . . .	151
1.1.2	The first sub-system fog . . . . .	151
1.2	The second sub-system . . . . .	152
2	The Digital Twin . . . . .	153
<b>V</b>	<b>Prospective Endeavors and Synopsis</b>	<b>155</b>
<b>6</b>	<b>Future Work</b>	<b>156</b>

<b>VI</b>	<b>General Conclusion</b>	<b>157</b>
<b>7</b>	<b>General Conclusion</b>	<b>158</b>

# List of Figures

2.1	A. IoT layered architecture three layered and B. five layered architecture. . . . .	13
2.2	Machine learning Types and Algorithms. . . . .	19
2.3	Source: Thomas Malone   MIT Sloan. . . . .	22
2.4	Decision Tree . . . . .	22
2.5	MLP with a Single Hidden Layer . . . . .	28
2.6	Simple Recurrent Neural Network . . . . .	30
2.7	Machine Learning VS Deep Learning . . . . .	31
2.8	Brief History of Digital Twins. . . . .	35
2.9	Development and spread of Digital Twins over time. . . . .	36
3.1	Digital Twin's Example Representation . . . . .	44
3.2	Digital Simulation VS Digital Shadow VS Digital Twin . . . . .	51
3.3	Digital Twin's types Example Robot Poppy Ergo Jr . . . . .	55
3.4	Sigmoid and Tanh Functions . . . . .	62
3.5	LSTM Forget Layer Operation . . . . .	63
3.6	LSTM Input Gate Layer Operation . . . . .	66
3.7	LSTM Cell State Operation . . . . .	67
3.8	LSTM Output Gate Operation . . . . .	68
3.9	Results on medium-sized datasets with only numerical features . . . .	69
3.10	Results on medium-sized datasets, with both numerical and categorical features . . . . .	70
3.11	Exemplary DT architecture. . . . .	78

4.1	Raspberry Pi 3 Model B . . . . .	86
4.2	GrovePi+. . . . .	88
4.3	Grove DHT11. . . . .	89
4.4	Grove - Barometer Sensor(High-Accuracy). . . . .	90
4.5	Grove - Light Sensor. . . . .	90
4.6	Grove-VOC and eCO2 Gas Sensor(SGP30). . . . .	91
4.7	The system's architecture in a simplified form. . . . .	95
4.8	The system's architecture in a detailed form. . . . .	98
4.9	The system's architecture in an elaborated form. . . . .	102
4.10	The Physical Twin's Edge Architecture. . . . .	104
4.11	The Physical Twin's Edge Architecture. . . . .	106
4.12	The System's Architecture in the Form of a Class Diagram . . . . .	109
4.13	Class "Component". . . . .	110
4.14	Class "Physical Twin". . . . .	110
4.15	Class "Physical Twin's Edge". . . . .	111
4.16	Class "Physical Twin's Sensor". . . . .	111
4.17	Class "Temperature Sensor". . . . .	112
4.18	Class "Humidity Sensor". . . . .	112
4.19	Class "Light Sensor". . . . .	113
4.20	Class "CO2 Sensor". . . . .	113
4.21	Class "Poppy Ergo Jr". . . . .	114
4.22	Class "Poppy Ergo Jr". . . . .	114
4.23	Class "PTFog". . . . .	115
4.24	Class "PTModel". . . . .	115
4.25	Class "CommunicationMedium". . . . .	115
4.26	Class "Digital Twin". . . . .	116
4.27	Class "DatasetAcquisition". . . . .	116
4.28	Class "Simulation". . . . .	117

4.29	Class "DTEdge". . . . .	118
4.30	Class "DTSensor". . . . .	118
4.31	Class "DTFog". . . . .	119
4.32	Class "DTModel". . . . .	119
4.33	Class "Perturbation". . . . .	120
4.34	Classes of Edge Perturbation". . . . .	120
4.35	External Dataset - Time Series Plot . . . . .	121
4.36	External Dataset - Correlation Matrix . . . . .	122
4.37	External Dataset - Temperature Histogram Plot . . . . .	123
4.38	External Dataset - Humidity Histogram Plot . . . . .	123
4.39	External Dataset - Light Histogram Plot . . . . .	124
4.40	External Dataset - CO2 Histogram Plot . . . . .	124
4.41	Global Dataset - Histogram Plots . . . . .	125
4.42	Physical Twin (PT) trained models on the external dataset . . . . .	127
4.43	The resulted Decision Tree trained on the external dataset . . . . .	128
4.44	Feature Importance Plot - Decision Tree trained on the external dataset	128
4.45	Feature Importance Plot - Random Forest trained on the external dataset . . . . .	129
4.46	Error Rate vs Number of Neighbors (K) . . . . .	130
4.47	Confusion Matrix - KNN trained on the external dataset . . . . .	131
4.48	Precision-Recall Curve - Naive Bayes trained on the external dataset	132
4.49	Confusion Matrix - Naive Bayes trained on the external dataset . . .	133
4.50	Loss Curve - MLP trained on the external dataset . . . . .	134
4.51	Confusion Matrix - MLP trained on the external dataset . . . . .	134
4.52	Feature Importance Plot - Decision Tree trained on the external dataset	136
4.53	Confusion Matrix - RNN trained on the external dataset . . . . .	136
4.54	Physical Twin's Selected Model . . . . .	138
4.55	Description of the Temperature Sensor in the Digital Twin . . . . .	140

4.56	DT trained models on the global dataset . . . . .	141
4.57	Decision Tree trained on the global dataset . . . . .	142
4.58	Feature Importance - Decision Tree trained on the global dataset . .	143
4.59	Confusion Matrix - Decision Tree trained on the global dataset . . .	143
4.60	Feature Importance - Random Forest trained on the global dataset .	145
4.61	Number of Neighbors - Decision Tree trained on the global dataset . .	146
4.62	Confusion Matrix - KNN trained on the global dataset . . . . .	147
5.1	Demonstration - PTEdge . . . . .	152
5.2	Demonstration - PTFog . . . . .	152
5.3	Demonstration - Poppy Ergo Jr . . . . .	153
5.4	Demonstration - Digital Twin . . . . .	153



# List of Tables

- 2.1 Key Differences Between Machine Learning and Deep Learning. . . . 32
- 3.1 Diverse Definitions of Digital Twins in Literature . . . . . 42
- 3.2 The Required and Optional Components of a Digital Twin. . . . . 45
- 3.3 The characteristics of the Digital Twin and their descriptions. . . . . 48
- 3.4 Diagnostic and Predictive Analytics VS Prescriptive Analytics . . . . 59
- 4.1 The Description of the External Dataset (Occupancy Dataset). . . . . 85

# List of Acronyms and Abbreviations

**DT** Digital Twin

**IDMU** Integral Digital Mock-Up

**NASA** National Aeronautics and Space Administration

**IoT** Internet of Things

**GE** General Electric

**QoS** Quality of Service

**AI** Artificial Intelligence

**ML** Machine Learning

**ITU** International Telecommunication Union

**NGN** Next-Generation Networks

**RFID** Radio-Frequency IDentification

**BLE** Bluetooth Low Energy

**PLM** Product Life-cycle Management

**DTP** Digital Twin Prototype

**DTI** Digital Twin Instance

**PT** Physical Twin

**CMMs** Coordinate Measuring Machine

**VVA** Verification, Validation and Accreditation

**BD** Big Data

**DL** Deep Learning

**CPSs** Cyber-Physical Systems

**IT** Information Technology

**MAPE-K** Monitor-Analyze-Plan-Execute over a shared Knowledge

**KPIs** Key Performance Indicators

**CDT** Cognitive Digital Twin

**UI** User Interface

**MES** Manufacturing Execution System

**ERP** Enterprise Resource Planning

**WMS** Warehouse Management System

**API** Application Programming Interface

**UCI** University of California, Irvine

**OS** Operating System

**NN** Neural Network

**KNN** K-Nearest Neighbour

“The Use of Cognitive Digital Twins on an IoT System for Edge Resilience and Anomaly Detection” Engineering Thesis

**SVM** Support vector machine

**PCA** Principal Component Analysis

**SVD** Singular Value Decomposition

**HMM** Hidden Markov model

**MLPs** Multilayer Perceptrons

**CNNs** Convolutional Neural Networks

**CNN** Convolutional Neural Network

**RNNs** Recurrent Neural Networks

**LSTM** Long Short-Term Memory

**CSDT** Cognitive Super-Digital Twin

**HI** Heat Index

**AMQP** Advanced Message Queuing Protocol

**MQTT** Message Queuing Telemetry Transport

**SoS** System of Systems

**JSON** JavaScript Object Notation

# Part I

## General Introduction

Chapter

1

# General Introduction

## Contents

1	Context . . . . .	3
2	Problem Statement . . . . .	3
3	Objectives . . . . .	4
4	Thesis Organization . . . . .	4

# 1 Context

Since its inception, and with the introduction of new technologies like 5G, Artificial Intelligence (AI), and edge computing, the IoT has undergone continuous development, becoming more powerful and capable of revolutionizing business operations. As the use of IoT devices in businesses has grown, the importance of enhancing their resilience and maintenance has become indispensable. These devices need to maintain optimal and secure functionality, even in the face of disruptions such as cyber-attacks, power outages, and physical damage. Given the increasing number of connected devices and the data they generate, it is crucial for businesses to have systems and processes in place to manage and sustain these devices.

One approach to enhance the resilience of IoT systems and prevent disasters is to design the infrastructure and industrial systems with redundancy and failover mechanisms. Redundancy involves incorporating backup devices, networks, or data centers into the system, ensuring that if one component fails, the system can continue to operate. However, implementing this approach necessitates a significant investment of financial and human resources. As an alternative to relying solely on redundancy, the use of DT has been employed.

## 2 Problem Statement

Maintenance and resilience play a crucial role and have high importance in the functioning of any system or asset, regardless of its complexity, whether it's a simple household appliance or a sophisticated manufacturing plant or transportation network. Effective maintenance practices are essential to ensure smooth operations, minimize downtime, mitigate the risk of failures, and improve overall efficiency. On the other hand, resilience refers to a system's ability to withstand disruptions or

shocks and quickly recover while preserving its essential functions and capabilities.

However, traditional maintenance methods are often reactive, focusing on fixing problems after they have already occurred rather than preventing them proactively. This reactive approach can be time-consuming, expensive, and not always effective in identifying and preventing failures, especially in intricate and interconnected systems. Additionally, the growing complexity of systems and equipment, along with the increasing demand for reliability and efficiency, adds further challenges to maintenance and resilience efforts.

### 3 Objectives

In this paper, we aim to:

- Explain and present the concepts and principles of DTs, which possess a highly adaptable nature to cater to the specific requirements of their intended use cases.
- Evaluate and compare different methodologies utilized in the development of DTs and their application in the realm of resilience.
- Introduce the framework that is presented as CSDT and its implementation.

### 4 Thesis Organization

The given work is organized into four chapters, which are outlined as follows:

- **Chapter 1 « General Introduction » :** In this first chapter, we provide a contextual analysis and explicate the problem statement and objectives of our study.
- **Chapter 2: « Background and Definition » :** This second chapter aims to provide a comprehensive overview of the various components which are



pertinent to the topic at hand.

- **Chapter 3: « State of the Art » :** This third chapter procures a comprehensive analysis of the various works examined, followed by a detailed discussion of each approach, culminating in a comparative study and synthesis.
- **Chapter 4: « Contribution » :** This part represents the culmination of the research conducted in this paper and serves as an input to the realized framework CSDT.
- **Chapter 5: « Conclusion » :** This final part marks the culmination of the research conducted and the future perspectives.

## Part II

# Background and Definitions

# Chapter 2

## Background

### Contents

1	IoT . . . . .	<b>9</b>
1.1	Introduction . . . . .	9
1.2	Definition of IoT . . . . .	9
1.3	Historical Evolution of IoT . . . . .	10
1.4	Characteristics of IoT . . . . .	11
1.5	IoT Architectures . . . . .	12
2	Internet of Things Resilience . . . . .	<b>16</b>
2.1	Introduction . . . . .	16
2.2	Definition of Resilience . . . . .	17
2.3	IoT resilience . . . . .	17
3	Machine Learning . . . . .	<b>18</b>
3.1	Definition of Machine Learning . . . . .	18
3.2	Types of Machine Learning . . . . .	18
3.3	Machine Learning Models . . . . .	21
4	Deep Learning . . . . .	<b>25</b>
4.1	Definition of Deep Learning . . . . .	25

4.2	Deep Learning Models . . . . .	25
5	Machine Learning VS Deep Learning . . . . .	<b>30</b>
6	Digital Twins . . . . .	<b>33</b>
6.1	Digital Twins History . . . . .	33

---

# 1 IoT

## 1.1 Introduction

In this section, various definitions, architectures, and components related to the IoT will be explored. IoT refers to a network where physical devices are interconnected and communicate with each other, exchanging data through the internet. These devices encompass a wide range of items, such as smartphones, laptops, sensors, cameras, and appliances, etc and it has the power to revolutionize numerous industries, including healthcare, transportation, manufacturing, and agriculture. Frequently, IoT devices are equipped with sensors that gather data, which can be analyzed to gain valuable insights and support decision-making processes.

## 1.2 Definition of IoT

IoT or Internet of things, the backbone of the Digital Age, the technology that has the power and potential to revolutionize numerous industries, has no standard definition yet. But here are a few definitions that has been collected:

- The term IoT is defined by the International Telecommunication Union (ITU) as a global infrastructure for the information society that enables the interconnection of different assets based on communication technologies. [1]. And in terms of a network that it is “Available anywhere, anytime, by anything and anyone” [2].
- ITU-T Study Group 13 leads the work of the ITU on standards for Next-Generation Networks (NGN) and future networks (ITU, SERIES Y, 2005). It has defined IoT as: “A global infrastructure for the information society, enabling advanced services by connecting physical and/or virtual things based on existing and evolving interoperable information.” [2]

- In [3], it has been defined as “a network of physical objects. The internet has transformed from being solely a network of computers into a vast network encompassing devices of all types and sizes. This includes vehicles, smartphones, home appliances, toys, cameras, medical instruments, industrial systems, and even animals. All connected ,all communicating and sharing information based on stipulated protocols in order to achieve smart reorganizations, positioning, tracing, safe and control and even personal real time online monitoring , online upgrade, process control and administration. ”

### 1.3 Historical Evolution of IoT

The IoT traces its origins back to the early days of the Internet, although it wasn’t officially defined until 1999 when Kevin Ashton<sup>1</sup> coined the term. However, ideas, research, and studies related to IoT had been present for some time prior to that.

Since its inception, the IoT has undergone continuous evolution, progressing through various stages of technological development and expanding into new domains of application. It has grown far beyond being a mere technology and has now become an integral part of our daily lives, enhancing efficiency and convenience [4].

In a study [5], it is mentioned that one of the earliest instances of an internet-connected device was a Coke machine at Carnegie Mellon University in the early 1980s. Programmers had the ability to connect to the machine via the internet and remotely monitor its status, check the availability of drinks, and even determine the temperature of the beverages.

This marked the beginning of the proliferation of interconnected devices. In the 1990s, the automotive industry pioneered the use of RFID technology to track inventory in factories and warehouses. By the early 2000s, there were already early

---

<sup>1</sup>Kevin Ashton is a British technology pioneer and his work in Radio-Frequency IDentification (RFID) technology and supply chain management paved the way for the development of the Internet of Things.

examples of IoT applications in sectors such as healthcare and logistics.

Presumably, before 2025, IoT will have a significant impact on daily life. IoT can be used in Electronic Voting, Electronic Identifications and in Medical Field to Support Patients. Robots are working in several sections using IoT. Remote sensing Robots are also Using IoT. IoT based systems are widely used in Farming. Remote sensing robots are collecting data with IoT protocols. [4]

## 1.4 Characteristics of IoT

The fundamental characteristics of the IoT are defined in [3] as follows:

**Interconnectivity:** The global information and communication infrastructure has the potential to interconnect anything.

**Things-related services:** The IoT has the potential to offer services pertaining to tangible objects while considering aspects such as privacy protection and aligning the virtual and physical aspects of these objects. To accomplish this, modifications are needed in both the technologies employed in the physical realm and the realm of information. These alterations are crucial for enabling the provision of services related to things while adhering to the limitations and demands imposed by physical objects.

**Heterogeneity:** The IoT devices exhibit heterogeneity because they operate on varying hardware platforms and networks, and utilize different networks to communicate with other devices or service platforms.

**Dynamic changes:** Devices within the IoT experience dynamic fluctuations in their states, including transitions between sleep and wake modes, connectivity and disconnection, and variations in contextual factors like location and speed. Furthermore, the number of devices within the network can also undergo dynamic fluctuations.

**Enormous scale:** The number of devices requiring management and intercommunication within the IoT is projected to be at least ten times greater than the

current number of devices connected to the Internet.

**Safety:** While harnessing the benefits of the IoT, it is vital to pay careful attention to safety issues. As both creators and beneficiaries of the IoT, we must prioritize safety considerations, which include protecting our personal information and physical well-being. To ensure comprehensive security, it is necessary to establish a scalable security framework that can effectively protect endpoints, networks, and the transmitted data.

**Connectivity:** The capability to establish connections enables easy accessibility and compatibility within a network. Accessibility refers to the ability to join a network, while compatibility refers to the capacity to exchange and utilize data in a standardized manner.

## 1.5 IoT Architectures

As mentioned, IoT has brought a significant change in the manner and the way we interact with physical objects and devices located in our environment. It facilitates communication and information exchange between them over the internet. Nevertheless, due to the vast number of devices involved and the complexity of the network infrastructure required to support them, designing and implementing IoT architecture is an essential starting point.

There are various and plenty types of IoT architectures, each with its unique advantages and challenges and depend fully on the corresponding use case and subject. The choice of architecture relies as well on several other factors beside specific use case, such as the network topology, the scalability, the reliability, and cost-effectiveness.

In this subsection, the most common IoT architectures are mentioned:



### 1.5.1 The Three and Five Layered Architectures

The article [6] talked about two architectures that are widely used: The three and five-layered architectures that are represented in Figure 2.1

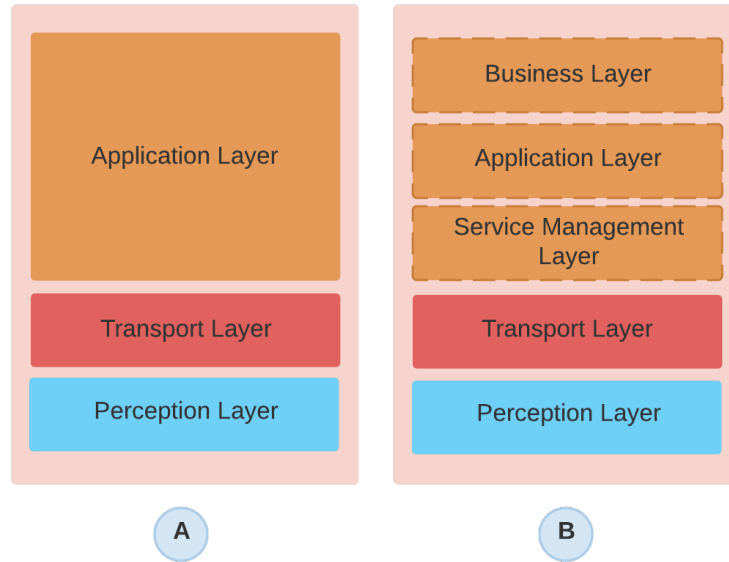


Figure 2.1: A. IoT layered architecture three layered and B. five layered architecture.

- **Perception Layer:** The perception layer is composed of a set of group of objects. These objects act as a bridge between the Physical world and the digital realm utilizing sensors to capture data. Its main objective is to gather information from the environment through a range of sensors for example temperature, humidity, light, CO2 sensors, cameras, etc., according to the specific use case and needs of the application. Researchers are primarily concerned with ensuring the proper identification, management, and security of these objects within this layer.
- **Transport Layer:** The purpose of this layer is to establish secure connections between objects and facilitate the sharing of information among them. Different communication protocols such as Ethernet, WiFi, Wi-MAX, ZigBee,

and Bluetooth Low Energy (BLE) can be used to enable this information exchange. However, there are still certain challenges that need to be addressed at this layer, such as reducing energy consumption in the network, ensuring Quality of Service (QoS), and adapting to dynamic topologies.

- **Service Management Layer:** It can also be referred to as "the Middleware Layer", this level facilitates and enables the integration of diverse and heterogeneous devices into IoT applications. Additionally, Moreover, it plays a pivotal role in processing raw data collected by objects in the perception layer. This data is typically characterized by its large volume and diverse nature.
- **Application Layer:** The layer in question is primarily tasked with providing application-specific and use case-specific services to end-users, that is why it plays a significant role in enhancing the convenience, safety and overall quality of life of end-users. However, the ability to tailor services to meet specific needs and preferences makes this layer critical in the success and adoption of IoT applications. As such, developers and researchers must continually work to identify and address the unique challenges associated with providing application-specific services in this layer.
- **Business Layer:** The business layer serves as the supervisor of an IoT system's operations and services, utilizing raw data acquired from other layers to create flow charts, graphs, and business models. Additionally, this layer is responsible for monitoring, analyzing, and evaluating the IoT system and its related components. Decision-making is a central activity of the business layer, as it plays a critical role in determining the direction and success of the IoT system.

### 1.5.2 The Edge Fog Cloud

Edge, fog, and cloud computing are different types of data storage and management in IoT.

Edge computing refers to computation at the edge of a device's network, while fog computing is an extension of cloud computing that acts as a layer between the edge and the cloud.

Fog computing is designed to overcome the challenges of edge computing, such as delays in detection, by processing data in real-time.

The cloud, on the other hand, refers to the on-demand delivery of IT services/resources over the internet.

Down below each layer is explained taken from [7].

- **Edge:**

Edge computing involves processing data locally within the network, specifically on edge devices and gateways, instead of relying on centralized storage. By avoiding data transfer to the cloud, it enables quick response times and unmatched speed.

When it comes to decentralized storage, edge computing stands as the most secure option. Unlike cloud storage, which distributes data across numerous servers, edge computing employs a vast number of local nodes, potentially reaching into the thousands. Each device within the edge network can function as an independent server, making it extremely difficult for hackers to breach. Gaining synchronized access to thousands of dispersed devices is practically unattainable.

This distinction also sets fog computing apart from edge computing. Fog computing serves as a network that connects to the cloud, while edge devices operate with loose connections and have the ability to act autonomously.

- **Fog:**

Fog computing serves as an intermediary layer positioned between the conventional centralized data storage system (cloud) and edge devices. Its purpose is to extend the capabilities of the cloud by bringing computation and data storage closer to the edge. Fog encompasses multiple nodes, known as fog nodes, forming a decentralized ecosystem—this stands as the primary contrast between fog and cloud computing.

When data reaches the fog layer, the individual node determines whether to process it locally or transmit it to the cloud. Consequently, the data remains accessible even offline since certain portions of it are stored locally. This presents another significant divergence between fog computing and cloud computing, as the latter relies on remote servers to execute and store all the intelligence and computations.

- **Cloud:**

It is a centralized storage situated further from the endpoints than any other type of storage. This explains the highest latency, bandwidth cost, and network requirements. On the other hand, cloud is a powerful global solution that can handle huge amounts of data and scale effectively by engaging more computing resources and server space. It works great for big data analytics, long-term data storage and historical data analysis.

## 2 Internet of Things Resilience

### 2.1 Introduction

Resilience is the ability of a system to recover from disruptions and continue to function effectively. In the context of technology, resilience is becoming increasingly important as our reliance on interconnected systems grows. Disruptions such as

cyber-attacks, natural disasters, and equipment failures can have significant consequences for businesses and individuals. Resilient systems are designed to minimize the impact of such disruptions and ensure continuity of service.

## **2.2 Definition of Resilience**

Resilience can be defined as the capacity to adapt, adjust and the ability of a system to continue operating and delivering services in the face of various and different types of failures or disruptions, such as hardware or software failures, network outages, cyber attacks, or natural disasters. Resilient computer systems are designed to anticipate and withstand these challenges, and to recover quickly and efficiently in the event of a failure or disruption by implementing redundancy, fault tolerance, and disaster recovery mechanisms, as well as conducting regular testing and maintenance to ensure that the system remains robust and reliable. Resilience is a critical attribute of modern computer systems, particularly those that are used to provide essential services or support critical business operations.

## **2.3 IoT resilience**

In the context of IoT, resilience refers to the capability of IoT systems to maintain reliable and secure connectivity, data transmission, and functionality in the face of various challenges and disruptions, such as network congestion, hardware failures, cyber attacks, or power outages.

It also refers to the ability of IoT systems to resist perturbances, recover from emergencies, and continue functioning in the face of disruptions. There are several scientific efforts to make IoT systems resilient, and AWS IoT Core features data redundancy and specific features for data resiliency, such as device shadow and AWS IoT Device Advisor. However, AWS IoT Core resources are region-specific and not replicated across regions unless specifically done so. Resilience is increasingly

important as IoT becomes a critical part of the global internet [8] [9].

## **3 Machine Learning**

### **3.1 Definition of Machine Learning**

Machine learning is a branch of AI and computer science that focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy [10]. It allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so.

It holds significance as it provides enterprises with insights into customer behavior trends and operational patterns, facilitating the creation of innovative products. Prominent companies like Facebook, Google, and Uber have embraced ML as a fundamental aspect of their operations, establishing it as a crucial factor for gaining a competitive edge [11].

### **3.2 Types of Machine Learning**

ML algorithms can be broadly classified into three types:

- Supervised Learning.
- Unsupervised Learning.
- Reinforcement Learning.

Figure 2.2 represents a diagram that illustrates the different ML algorithm, along with the categories.

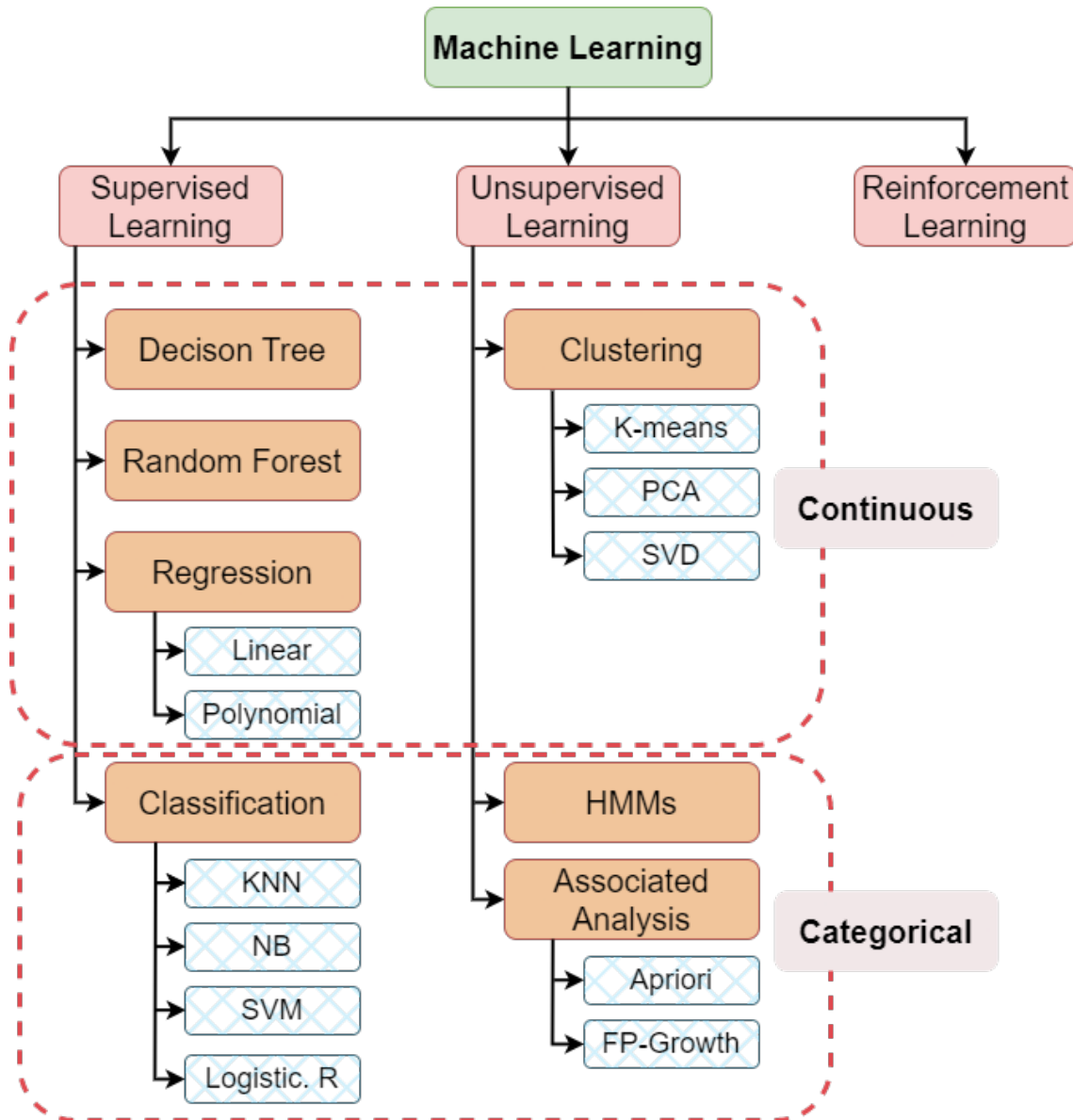


Figure 2.2: Machine learning Types and Algorithms.

### 3.2.1 Supervised Learning

Supervised learning is a category within ML that relies on external guidance for the machine to learn [12]. In supervised learning, models are trained using labeled datasets [10]. Following training and processing, the model is evaluated by providing it with sample test data to determine if it accurately predicts the desired output.

The objective of supervised learning is to establish a mapping between input data and output data. It mirrors the concept of a student learning under the super-

vision of a teacher. An example of supervised learning is spam filtering.

Supervised learning can be further categorized into two types of problems:

- Classification.
- Regression.

### **3.2.2 Unsupervised Learning**

Unsupervised learning is an algorithmic approach in ML that examines and clusters datasets lacking pre-existing labels to uncover patterns and insights [10].

Unlike supervised learning, unsupervised learning does not rely on a training dataset to guide the models. Instead, the models autonomously discover concealed patterns and group the data based on similarities and dissimilarities [13].

Unsupervised learning is employed to unveil the underlying structure of datasets and finds applications across diverse domains, aiding in data feature summarizing and explanation.

Additionally, it serves as a means of testing AI and is capable of performing more intricate processing tasks compared to supervised learning systems [10].

Hence further, it can be classified into two types:

- Clustering.
- Association.

Examples of some Unsupervised learning algorithms are K-means Clustering, Apriori Algorithm, Eclat, etc.



### 3.2.3 Reinforcement Learning

Reinforcement Learning is a form of ML that allows an agent to learn within an interactive environment through trial and error, utilizing feedback obtained from its own actions and experiences [14].

This approach revolves around rewarding desired behaviors and penalizing undesired ones.

The primary focus of reinforcement learning is determining how intelligent agents should take actions in an environment to maximize cumulative rewards [15].

Reinforcement learning algorithms acquire knowledge from outcomes and make decisions about the subsequent actions to be taken. It has demonstrated successful applications in various domains, such as robot control, elevator scheduling, telecommunications, backgammon, checkers, and Go.

Reinforcement learning serves as a valuable technique for automated systems seeking to identify the optimal behavior or path in specific situations. Q-Learning algorithm is used in reinforcement learning.

Figure 2.3<sup>2</sup>, realized by Thomas Malone, represent the way on what Machine Learning models can perform.

## 3.3 Machine Learning Models

### 3.3.1 Decision Trees

The decision tree is a supervised learning algorithm primarily employed for solving classification problems, although it can also tackle regression problems. It accommodates both categorical and continuous variables [12].

The decision tree presents a tree-like structure comprising nodes and branches

---

<sup>2</sup>See: <https://bit.ly/3gvRho2>

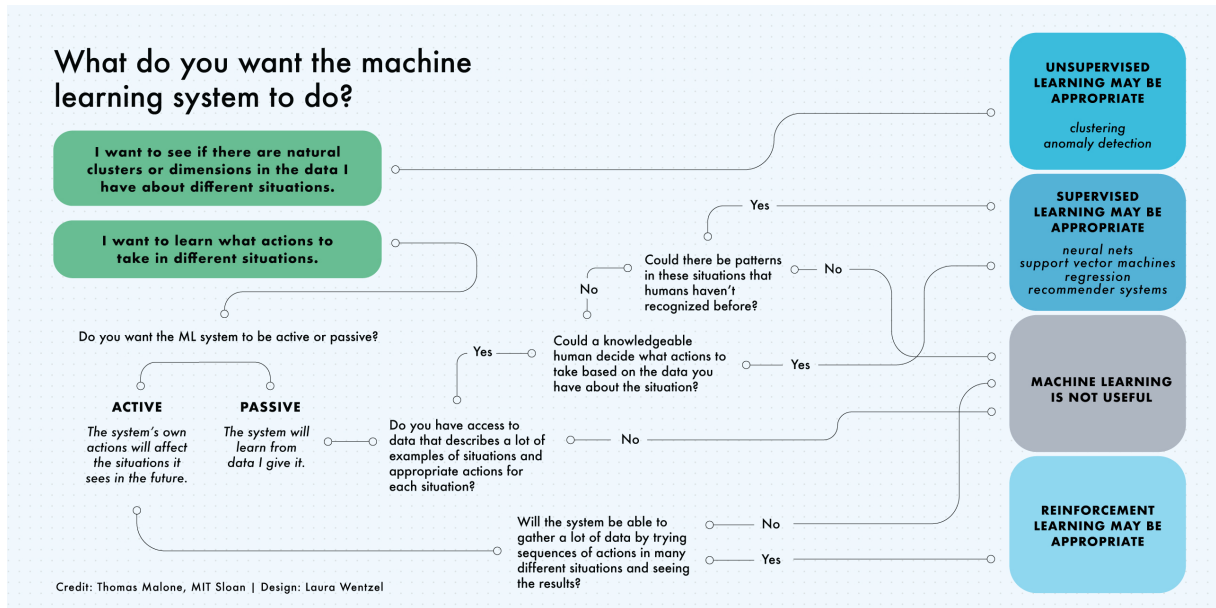


Figure 2.3: Source: Thomas Malone | MIT Sloan.

to represent decisions and their possible consequences (Figure 2.4). It initiates with a root node and further branches out to leaf nodes. Internal nodes represent dataset features, branches denote decision rules, and leaf nodes signify the problem's outcomes [16].

Decision tree algorithms find practical application in various real-world scenarios. For instance, they are utilized in distinguishing between cancerous and non-cancerous cells and providing car purchase recommendations to customers.

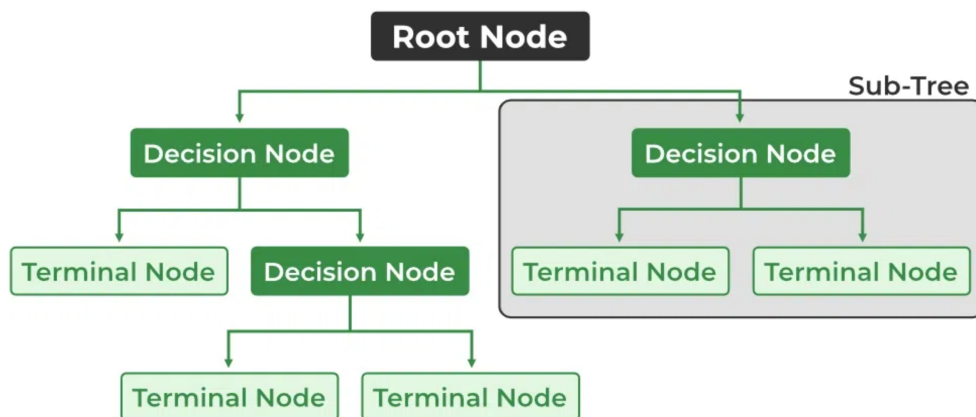


Figure 2.4: Decision Tree

### 3.3.2 Random Forests

Random forest is a supervised learning algorithm employed in ML for both classification and regression tasks. It operates as an ensemble learning technique, leveraging multiple classifiers to generate predictions and enhance the model's performance [12].

This approach encompasses numerous decision trees that operate on subsets of the provided dataset, amalgamating their outcomes to improve predictive accuracy. It is recommended to have a random forest consisting of 64 to 128 trees, as a higher number of trees typically leads to increased algorithmic precision.

In other words, The fundamental concept behind random forest is the wisdom of crowds. A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models. It forest uses bagging and feature randomness when building each tree to ensure that the trees are uncorrelated [17].

When classifying a new dataset or object, each tree produces a classification result, and the algorithm predicts the final output based on majority voting.

Random forest demonstrates efficient processing capabilities, making it suitable for handling missing and inaccurate data. Additionally, it offers a swift execution speed.

### 3.3.3 Naive Bayes

The Naive Bayes classifier is a supervised learning algorithm utilized for making predictions by considering the probability of an object. It derives its name from Bayes theorem, as it follows the assumption that variables are independent of each other, hence "naïve." [12]

Naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable.

Despite their naive design and oversimplified assumptions, Naive Bayes classifiers

have worked well in many complex real-world situations [18].

Bayes theorem, on which this algorithm is based, deals with conditional probability. It calculates the likelihood of event A occurring given that event B has already taken place. The equation for Bayes theorem is expressed in Equation (2.1).

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \quad (2.1)$$

Naïve Bayes classifier is one of the best classifiers that provide a good result for a given problem. It is easy to build a naïve bayesian model, and well suited for the huge amount of dataset. It is mostly used for text classification.

### 3.3.4 K-Nearest Neighbors

The K-Nearest Neighbour (KNN) algorithm is a supervised learning technique applicable to both classification and regression problems. It operates by establishing the similarities between a new data point and existing data points. Utilizing these similarities, the algorithm categorizes the new data point into the most similar class. It is also referred to as a "lazy learner" algorithm because it retains all available datasets and classifies each new instance with the assistance of its K-nearest neighbors.

To assign the new instance to the most similar class, KNN calculates the distance between data points using a distance function. Common distance functions include Euclidean, Minkowski, Manhattan, or Hamming distance, chosen based on the specific requirements of the problem [12].

## 4 Deep Learning

### 4.1 Definition of Deep Learning

DL, a subset of ML, employs ANN comprising multiple layers to extract high-level features from raw input data. It mimics the human learning process and is considered a form of AI.

The algorithms used in DL are organized hierarchically, with each layer growing in terms of complexity. They find application in various tasks, including supervised and unsupervised learning, such as speech recognition, image classification, and natural language processing. Deep learning plays a crucial role in data science, which encompasses statistics and predictive modeling, offering significant advantages to data scientists responsible for gathering, analyzing, and interpreting large volumes of data.

### 4.2 Deep Learning Models

Several DL algorithms are widely used, including Multilayer Perceptrons (MLPs), CNNs, Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), Radial Function Networks, and Self-Organizing Maps.

MLPs are considered the fundamental and oldest deep learning algorithm. CNNs are particularly effective for image and video recognition tasks, while RNNs and LSTMs are commonly employed for natural language processing and speech recognition. Radial Function Networks and Self-Organizing Maps are utilized for clustering and classification purposes.

As mentioned before, DL algorithms are designed to run dynamically through multiple layers of NN, with pre-training specifically tailored to the given task.

A few of the cited models are about to be presented down below:

#### 4.2.1 CNNs

A Convolutional Neural Network (CNN) is a widely utilized neural network architecture in the realm of AI's Computer Vision field [19]. Popular type of neural network architecture used in the field of Computer Vision within Artificial Intelligence. Computer vision enables computers to interpret and understand visual data, such as images. In the realm of Machine Learning, Artificial Neural Networks exhibit strong performance. They are employed in various datasets encompassing images, audio, and text. Different types of Neural Networks serve different purposes. For instance, Recurrent Neural Networks, particularly Long Short-Term Memory (LSTM) networks, are suitable for predicting word sequences, while Convolutional Neural Networks are commonly used for image classification. A typical Neural Network consists of three types of layers:

- **Input Layers:** This initial layer receives the input data for the model. The number of neurons in this layer is equivalent to the total number of features in the data (e.g., the number of pixels in an image).
- **Hidden Layers:** The input from the Input layer is transmitted to the hidden layer(s). The number of hidden layers can vary depending on the model and the size of the data. Each hidden layer may contain a different number of neurons, typically exceeding the number of features. The output of each layer is computed by performing matrix multiplication between the output of the previous layer, which has learnable weights, and subsequently adding learnable biases. This is followed by an activation function, which introduces nonlinearity to the network.
- **Output Layer:** The output from the hidden layer is fed into a logistic function, such as sigmoid or softmax, which converts the output of each class into a probability score for that class. The data is fed into the model, and the output from each layer is obtained through a process called feedforward. Subsequently, the

error is calculated using an error function, such as cross-entropy or square loss error. The error function measures the performance of the network. The next step involves backpropagation, where derivatives are calculated to minimize the loss. Backpropagation is essential for adjusting the model's parameters and improving its performance.

Convolutional Neural Network consists of multiple layers like the input layer, Convolutional layer, Pooling layer, and fully connected layers. The Convolutional layer applies filters to the input image to extract features, the Pooling layer down-samples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

#### **4.2.2 Multilayer Perceptrons**

MLP serves as an extension of the feed-forward neural network. It encompasses three distinct layers as depicted in

the input layer, the output layer, and hidden layers, as depicted in Figure 2.5:

1. Input Layer.
2. Output Layer.
3. Hidden Layers.

The input layer receives the input signal for processing, while the output layer is responsible for performing tasks such as prediction and classification.

The true computational engine of the MLP resides within an arbitrary number of hidden layers positioned between the input and output layers. Similar to a feed-forward network, the data flows in a forward direction from the input layer to the output layer [20].

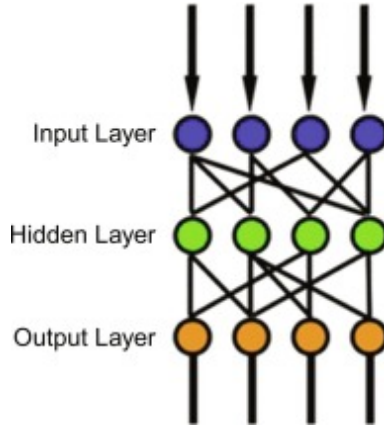


Figure 2.5: MLP with a Single Hidden Layer

In an MLP, the neurons are trained using the backpropagation learning algorithm. MLPs are specifically designed to approximate any continuous function and can effectively address problems that are not linearly separable. Prominent use cases of MLPs include pattern classification, recognition, prediction, and approximation.

#### 4.2.3 Recurrent Neural Networks

An ANN known as a RNN is specifically designed for handling sequential or time-series data.

Unlike CNN, RNNs incorporate hidden states and allow the utilization of previous outputs as inputs. This enables RNNs to effectively process sequential data by utilizing the output from one time step as the input for the next step. RNNs find extensive applications in various fields such as natural language processing and speech recognition. Nonetheless, RNNs do have certain drawbacks, including challenges with training due to issues like the vanishing and exploding gradients [21].

Another constraint of traditional RNNs is the lack the ability to incorporate future inputs into the current state. Furthermore, RNNs encounter difficulties when dealing with long-term dependencies, which can result in problems such as gradient vanishing and exploding.



However, a solution to these limitations emerged in the form of Long Short-Term Memory Networks (LSTMs). LSTMs were introduced to address these shortcomings by enabling the learning of long-term dependencies through the retention of information over extended periods [22].

### **1. Long Short Term Memory (LSTM):**

LSTM is a specific type of ANN that finds application in DL and ML tasks. It serves as a variation of RNNs and exhibits the capability to effectively handle lengthy time-series data, enabling the learning of order dependencies in sequence prediction tasks.

In contrast to conventional feedforward neural networks, LSTM incorporates feedback connections and possesses the ability to process not only individual data points but also complete data sequences.

One of the primary objectives of LSTM is to address the challenge of long-term dependencies encountered by RNNs. While RNNs struggle to predict information stored in long-term memory, LSTM provides more accurate predictions by leveraging recent information. The structure of LSTM consists of a chain comprising four neural networks and incorporates memory blocks known as cells. These cells retain information, and the manipulation of memory is facilitated by specialized components called gates [23].

Each recurrent neural network consists of a series of repeating neural network modules, forming a chain. These networks incorporate loops, allowing information to be retained within the network. Figure 1 illustrates a simple recurrent neural network with loops. In this figure, the neural network denoted as Figure 1, A takes the input  $x_t$  and generates the output  $h_t$ . The presence of a

loop facilitates the transfer of data from one phase of the network to the next. LSTM is explicitly designed to tackle the problem of long-term dependencies. Each recurrent neural network is composed of a sequence of repeating neural network modules. To aid in comprehension of the subsequent sections, Table 2 presents a list of symbols that are utilized to explain the various concepts [23] [24].

Figure 2.6 illustrates a simple recurrent neural network with loops. LSTM takes the input  $x_t$  and generates the output  $h_t$ . The presence of a loop facilitates the transfer of data from one phase of the network to the next. LSTM is explicitly designed to tackle the problem of long-term dependencies. Each recurrent neural network is composed of a sequence of repeating neural network modules [24].

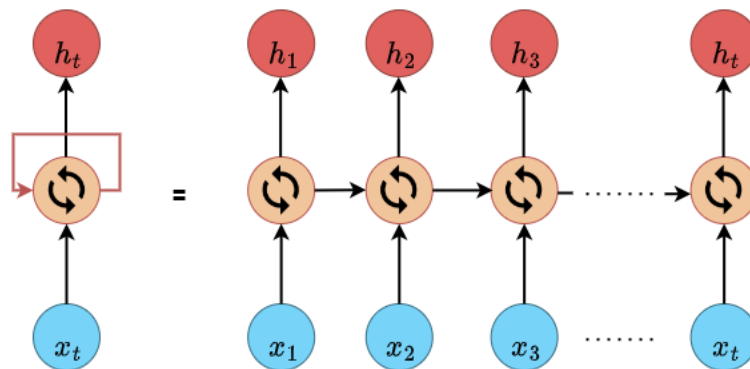


Figure 2.6: Simple Recurrent Neural Network

## 5 Machine Learning VS Deep Learning

Taking into account Figure 2.7:

- The functioning of ML models can be illustrated through the example of image recognition for distinguishing between cats and other animals.

In this scenario, the ML model takes images of cats as input. It then extracts distinct features from these images, such as shape, height, nose, eyes, and other relevant characteristics. By employing a classification algorithm, the model analyzes these features and generates a prediction as output.

- The functioning of DL can be comprehended using the same example of distinguishing mentioned previously.

In DL models, the images serve as input and are directly fed into the algorithms, eliminating the need for manual feature extraction. The images traverse through various layers of an artificial neural network, allowing the model to predict the final output.

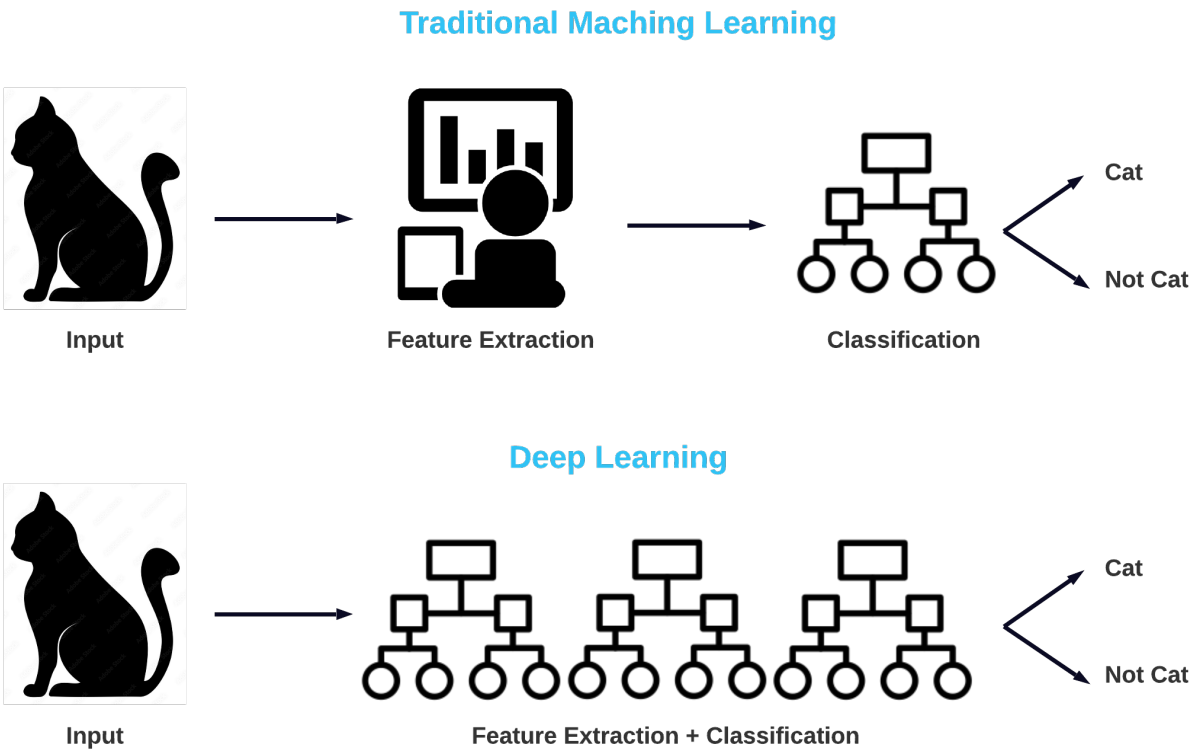


Figure 2.7: Machine Learning VS Deep Learning

Table 2.1 taken from [25] shows the Key comparisons between ML and DL.

Parameter	ML	DL
<b>Data Dependency</b>	Although ML depends on the huge amount of data, it can work with a smaller amount of data.	DL algorithms highly depend on a large amount of data, so we need to feed a large amount of data for good performance.
<b>Execution Time</b>	ML algorithm takes less time to train the model than DL, However, testing the model can be time-consuming and requires a significant duration.	DL takes a long execution time to train the model, but less time to test the model.
<b>Hardware Dependencies</b>	Since ML models do not need much amount of data, so they can work on low-end machines.	The DL model needs a huge amount of data to work efficiently, so they need GPU's and hence the high-end machine.
<b>Feature Engineering</b>	ML models need a step of the interaction with the expert performing feature extraction, after which it continues to progress.	DL is the enhanced version of ML, so it does not need to develop the feature extractor for each problem; the problem-solving approach focuses on allowing the model to learn high-level features directly from the data.
<b>Problem-solving approach</b>	To solve a given problem, the traditional ML model breaks the problem in sub-parts, and after solving each part, produces the final result.	The problem-solving approach of a DL model is unlike traditional ML models.
<b>Interpretation of result</b>	The ease of interpreting the result for a specific problem is evident. As when we work with ML, we can interpret the result easily, it means why this result occur, what was the process.	The interpretation of the result for a given problem can get very difficult. We may get a better result for a given problem than the ML model, but we cannot find why this particular outcome occurred, and the reasoning.
<b>Type of data</b>	ML models mostly require data in a structured form.	DL models can work with structured and unstructured data both as they rely on the layers of the ANN.
<b>Suitable For</b>	ML models are suitable for solving both simple and moderately complex problems.	DL models are suitable for solving complex problems.

Table 2.1: Key Differences Between Machine Learning and Deep Learning.

## 6 Digital Twins

### 6.1 Digital Twins History

The article [26] recorded in 2019 that over 850 academic papers on the topic of Digital Twins have been published since 2016.

The concept of a "twin" has its roots in the National Aeronautics and Space Administration (NASA) Apollo program of the 1970s. During this time, NASA built a replica of space vehicles on Earth that mimicked the equipment's condition during the mission. This was done to ensure that NASA could test and prepare for every possible scenario that might occur during the mission. This was the first application of the "twin" concept [27].

In 2003, Michael Grieves, a professor of engineering at the University of Michigan, proposed the idea of a DT in his Product Life-cycle Management (PLM) course. DT is a virtual digital representation of physical products that can be used to simulate and analyze real-world scenarios in a virtual environment. DT technology enables manufacturers to create a digital copy of a physical product, which can then be used to monitor and predict its performance, optimize its design, and reduce the time and cost of maintenance and repairs.

In 2012, NASA applied DT to integrate high-fidelity simulation with a vehicle's on-board health management system, maintenance history, and fleet data to mirror the life of its flying twin. This allowed NASA to monitor the health and performance of their equipment in real-time, identify potential problems before they occurred, and increase safety and reliability.

The development of the IoT has boosted the manufacturing industry's adoption of DT technology. With the IoT, manufacturers can connect their physical

products to the internet and collect data on their performance in real-time. This data can then be used to create a DT of the product, which can be used to monitor and optimize its performance, predict maintenance needs, and improve its design.

Enterprises like Siemens and General Electric (GE) <sup>3</sup>, have developed DT platforms for real-time monitoring, inspection, and maintenance. These platforms enable manufacturers to monitor their products in real-time, identify potential problems before they occur, and reduce the time and cost of maintenance and repairs.

In 2017, Tao and Zhang proposed a five-dimensional DT framework to guide the digitalization and intellectualization of the manufacturing industry. The framework provides theoretical guidance for the digitalization and intellectualization of the manufacturing industry and includes five dimensions: physical, cyber, human, virtual, and knowledge.

From 2017 to 2019, Gartner continuously ranked DT among the top 10 technological trends with strategic values. DT is becoming increasingly important in the manufacturing industry, as it enables manufacturers to monitor and optimize their products in real-time, predict maintenance needs, reduce the time and cost of maintenance and repairs, and improve their designs.

Figure 2.8 provides a brief History summary of the DT.

Similar to what was previously stated, since their inception, DT have rapidly evolved and become increasingly popular in various industries such as manufacturing, healthcare, and urban planning, among others, thanks to their

---

<sup>3</sup>GE is a multinational conglomerate that operates in various industries including aviation, healthcare, renewable energy, and power generation. It was founded in 1892 and is based in Boston.

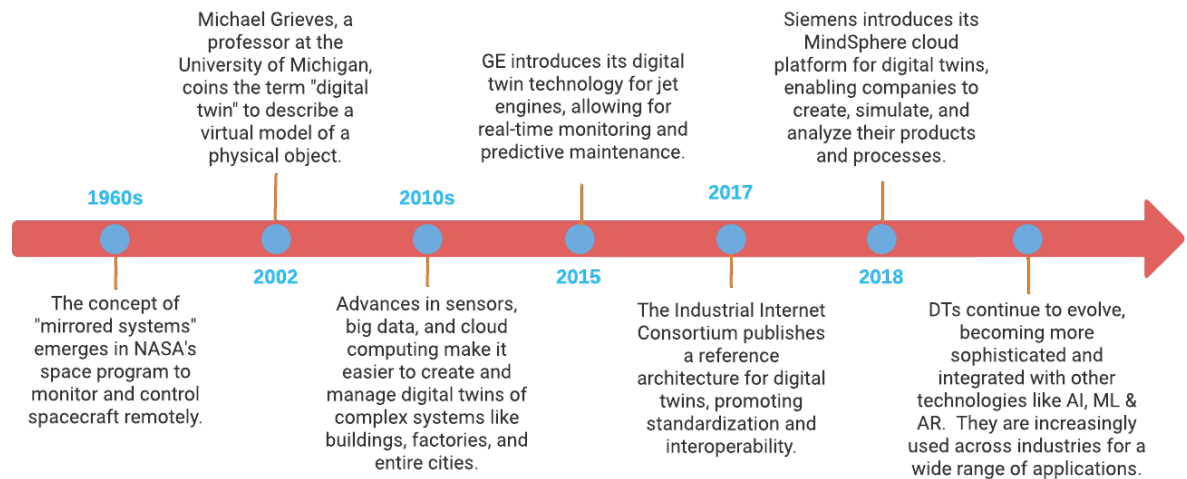


Figure 2.8: Brief History of Digital Twins.

ability to replicate real-world objects, processes, or systems with a high level of accuracy, simulate and test different scenarios, monitor their performance in real-time, and optimize their design and operation, thereby enabling organizations to make more informed decisions, improve their efficiency, reduce costs, and enhance their customer experience, and as technology advances and more data is collected, analyzed, and shared, it is likely that digital twins will continue to play a vital role in shaping the future of many sectors and transforming the way we live, work, and interact with the world around us.

Figure 2.9, taken from the article [26], shows advancement, evolution and development of DT over time.

This new technology is going to be discussed further more in Chapter 3.

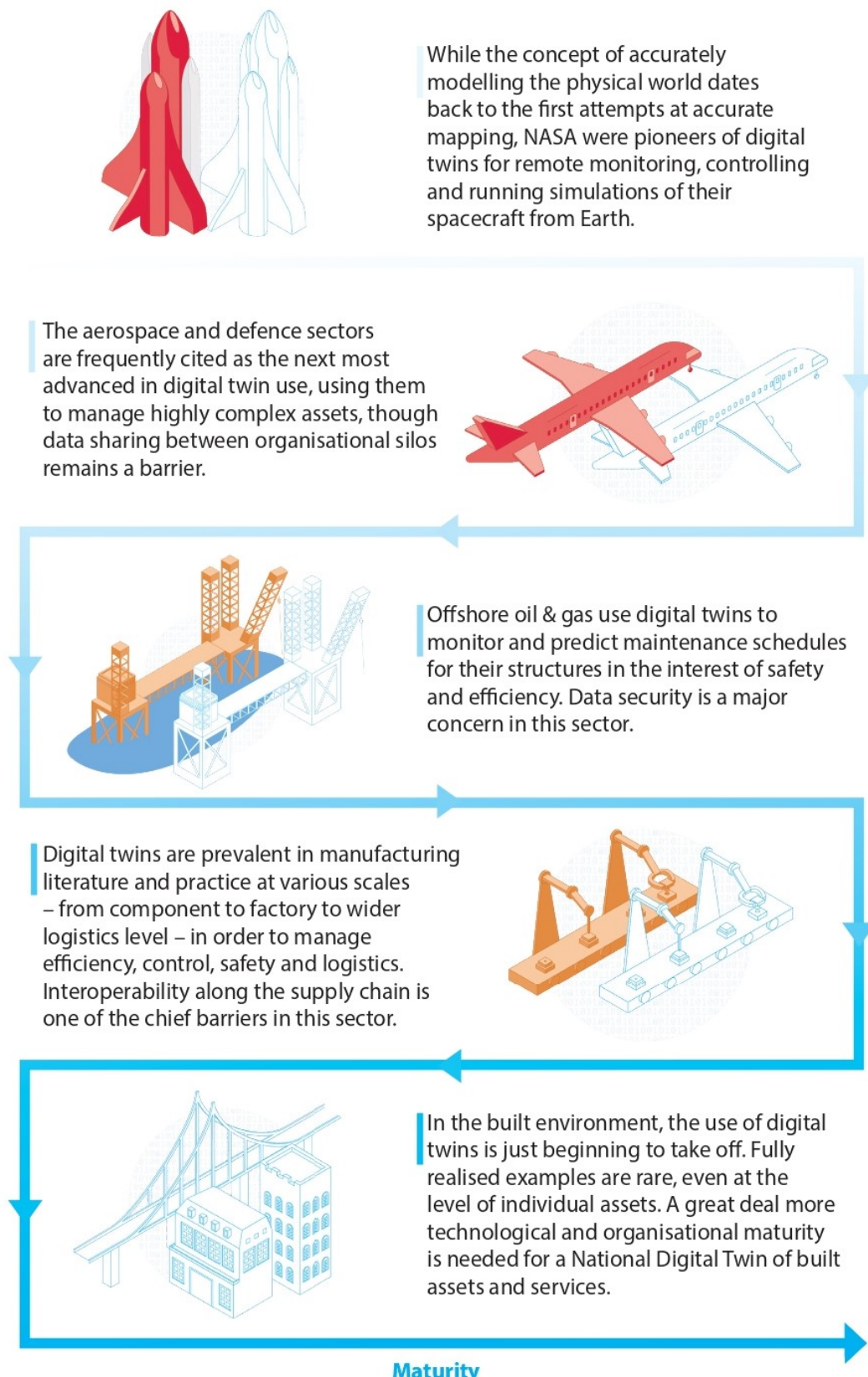


Figure 2.9: Development and spread of Digital Twins over time.



## Part III

### State of the Art

Chapter

3

# State of the Art

## Contents

1	Digital Twins Concepts . . . . .	<b>40</b>
1.1	Introduction . . . . .	40
1.2	Definition of Digital Twins . . . . .	40
1.3	Components of a Digital Twin . . . . .	43
1.4	Characteristics and Requirements of a Digital Twin	46
1.5	An Overview on the Predecessors of Digital Twins and Their Key Differences . . . . .	49
1.6	Different Types of Digital Twins . . . . .	52
2	Digital Twins with ML and DL . . . . .	<b>56</b>
2.1	Introduction . . . . .	56
2.2	ML Appliance in Digital Twins . . . . .	56
2.3	Selecting an Adapted Model for IoT tabular Data	59
3	The use of Digital Twins for Resilience and Prevention . .	<b>70</b>
3.1	Digital twins as run-time predictive models for the resilience of cyber-physical systems: a conceptual framework . . . . .	70

3.2	Cognitive Digital Twins for Resilience in Production: A Conceptual Framework . . . . .	73
3.3	State of the Art in using Digital Twins for prevention	74
4	Digital Twins architecture . . . . .	<b>76</b>
5	Framework CSDT . . . . .	<b>78</b>
5.1	Conclusion . . . . .	79

---

# 1 Digital Twins Concepts

## 1.1 Introduction

In recent years, the significance of digital twins has grown substantially within the IoT field. Essentially, a digital twin refers to a virtual representation of a physical object or system that is constructed using data obtained from sensors and other relevant sources.

Within the realm of IoT, digital twins find utility in real-time monitoring and management of physical assets and systems like buildings, vehicles, and manufacturing equipment. By simulating the behavior of the corresponding physical object or system, digital twins contribute to the identification of potential issues and the optimization of performance.

This chapter delves into a comprehensive exploration of the digital twin concept, providing a more detailed overview in the form of a state-of-the-art analysis. It covers the general definition of digital twins, their inherent characteristics, and their architectural aspects.

## 1.2 Definition of Digital Twins

DT faces challenges due to the lack of a universally accepted definition and established implementation standards. This lack of consensus makes it challenging to design, implement, and widely adopt this technology [28]. Furthermore, since DT is applied in various domains and relies on evolving technologies, it requires customization for each specific domain and is influenced by the current state of these technologies.

Table 3.1 displays a range of DT definitions along with their corresponding

reference and applied fields.

Numerous articles have focused on the absence of a fixed and pre-established concept for DTs. This gap has been addressed in several articles, which have proposed the following definitions:

- Grieves and Vickers define the DT as a connection of virtual and digital representations that comprehensively depict and describes the existing physical asset, encompassing its molecular composition and overall geometry. When functioning optimally, a Digital Twin provides all the information that would typically be gleaned from examining the physical counterpart. There are two types of Digital Twins: Digital Twin Prototype (DTP) and Digital Twin Instance (DTI) [28].
- "Various terms have been given in multiple literature works, such as 'ultra-high fidelity', 'cradle-to-grave', 'integrated' model , Integral Digital Mock-Up (IDMU). These terms are important and relevant to the DT concept, however, having multiple definitions and terms has delayed reaching a consensus on a single representative, unifying definition. In the simplest words, a digital twin is a 'digital' 'twin' of an existing physical entity" [28].
- "A DT is the virtual digital representation equivalent to physical products" [36].

### **1.2.1 Deducing a General Definition of Digital Twins**

In the course of exploring the literature on DT, it becomes apparent that many articles have examined the concept of DT within a particular domain, such as manufacturing or healthcare, as a result, a comprehensive and universally applicable definition of DT has been elusive.

Domain	Definition	
Aerospace	<ul style="list-style-type: none"> <li>- A DT is an integrated multiphysics, multiscale, probabilistic simulation of an as-built vehicle or a system that uses the state-of-the-art physical models and other relevant information to accurately replicate the life and behavior of its corresponding flying counterpart. The DT is ultra-realistic and may consider one or more important and interdependent vehicle systems.</li> <li>- DT is a life management and certification paradigm whereby models and simulations consist of as-built vehicle state, along with recorded loads, environmental conditions, and specific historical data related to the vehicle, in order to facilitate detailed and precise modeling of individual aerospace vehicles throughout their operational lifespan.</li> </ul>	[29] [30]
Industry	DT is an evolving digital profile of the historical and current behavior of a physical object or process that helps optimize business performance. It is based on massive, cumulative, real-time, real-world data measurements across an array of dimensions.	[31]
Engineering	A DT is a digital replica of physical assets, processes, and systems that can be used for various purposes, such as simulation, optimization, and monitoring.	[32]
Healthcare	A DT is a personalized, dynamic, and data-driven computational model that can be used to simulate an individual's physiology and health status, and to predict their response to treatment or changes in lifestyle.	[33]
Agriculture	DT is a dynamic approximation of an entity in virtual space, continuously updated through the collection of data, models, and what-if simulation. In the majority of applications found in current research, agricultural DT form a simplified or functionally reduced view of the observed entity or system, as cost, complexity, and goals are balanced with functionality and replication correctness requirements, as guided by the functional requirements of the intended application.	[34]
Manufacturing	DT are software models that represent the attributes and operating behavior of physical assets and processes. They support better decision making by simulating how assets behave given certain inputs.	[35]

Table 3.1: Diverse Definitions of Digital Twins in Literature

However, by synthesizing the information collected from these various sources, we can arrive at a global definition of DT that encompasses the most salient features and characteristics of the concept:

DT refer to a combination of virtual machines and computer-based models that enable the simulation, emulation, or mirroring of the behavior and characteristics of a physical entity, such as an object, a process, a human, or a human-related feature. The relationship between a DT and its PT is established through a bijective connection that enables continuous interaction, communication, and synchronization between the two.

Unlike static models or simulations, DT are living, intelligent, and evolving models that follow the life-cycle of their PT to monitor, control, and optimize their processes and functions. DT can predict future statuses, such as defects, damages, or failures, and simulate and test novel configurations to proactively apply maintenance operations.

The twinning process is facilitated by a closed-loop optimization approach that considers the DT, its PT, and the external surrounding environment. This approach ensures that DT are more than just simple models or simulations. They are a dynamic and responsive tool that allows designers, engineers, and operators to enhance the efficiency, safety, and performance of physical systems across various industries.

To aid in visualization, Figure 3.1 is provided.

### **1.3 Components of a Digital Twin**

The concept of digital twins was initially introduced by Grieves [37], who defined it as comprising three key components: the digital or virtual part, the

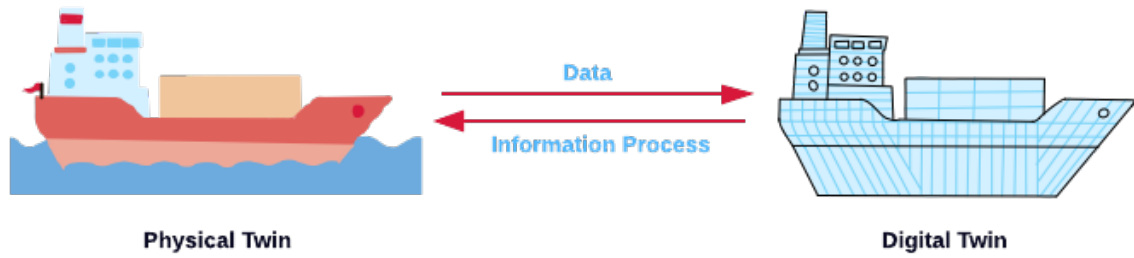


Figure 3.1: Digital Twin's Example Representation

physical product or asset, and the connection between them.

- On the virtual side, there has been significant improvement in the amount of available information. Additional behavioral characteristics have been incorporated, enabling not only visualization of the product but also the ability to test its performance capabilities.
- On the physical side, there is a greater capacity to gather information about the characteristics of the physical product. This includes collecting various physical measurements from automated quality control stations like Coordinate Measuring Machines (CMMs).

As the concept evolved, other authors, such as Tao et al., expanded the definition of a DT to include additional components like data and service. Tao et al. also recognized Verification, Validation and Accreditation (VVA) as essential elements of a digital twin. Miller et al., with the introduction of data models, further broadened the definition by incorporating the integration of multiple models [28].

Despite these efforts to refine the definition of a digital twin, achieving a consensus on its fundamental requirements remains challenging. This is due to variations in the necessary components and properties of digital twins across different works. Moreover, the domain-dependence of digital twins necessitates



defining components that can be universally applied across domains.

To tackle this challenge, researchers have compiled and integrated the essential components and properties from previous works to provide a comprehensive definition of a digital twin. These properties and components are considered necessary for the effective implementation and understanding of digital twins. By integrating the contributions of previous works, which have only been concerned with some components of DT, researchers aim to provide a more holistic definition of a DT.

Based on this analysis and understanding, researchers have defined the elementary and imperative components of a DT. These components provide a comprehensive definition of a DT that can be used across domains. By refining the definition of a DT and its fundamental requirements, researchers aim to provide a framework that can support the development and implementation of DT in a range of applications and industries.

The Table 3.2 taken from [28], summarises how each component contributes uniquely to the functions of DT. Removing any component voids the DT of the functionality and its uniqueness. The three first rows are required, and the rest contribute to the uniqueness of the Digital Twin.

Characteristics	Definitions
Physical Asset	What the digital twin is a twin of.
Digital Asset	The Digital Twin
Continuous Bijective Relation	For real-time synchronisation and twinning.
IoT	For data collection and information sharing.
Time Continuous Data	For synchronisation and input to ML.
ML	For analytics of the asset.
Security	To prevent data leaks and information compromises.
Evaluation metrics / Testing	To evaluate the performance of DT.

Table 3.2: The Required and Optional Components of a Digital Twin.

☞ **Note:** In this paper, the considered components are: Physical Asset, Digital Asset, Continuous Bijective Relation, IoT, ML.

## 1.4 Characteristics and Requirements of a Digital Twin

Although the definition of a digital twin may appear straightforward, it is the properties of the technology that distinguish it as more than just a mere digital replica. Some properties are required to create an accurate and authentic digital twin, while others are dynamic and can evolve over time. This section will explore both types of characteristics in detail.

### 1.4.1 Essential Characteristics (Requirements)

The Necessary properties and features mentioned in the Article [28] are:

- Real-time connection with the physical entity by making a bi-univocal relation between DT and the physical asset which means that the PT is uniquely paired with its DT.
- Self-evolution is a characteristic that has not been explored much. Self-evolution means that a DT can learn and adapt in real-time, by providing feedback to both physical asset and DT. This can be easily harnessed now due to the up rise of machine learning tools: to remodel and redesign itself (such as reinforcement learning). The frequency of this synchronisation depends on the update scenarios, such as event-based (supply chain), periodic intervals (aircraft), condition based (logistics), etc.
- Continuous ML analysis (dependent on the frequency of the synchronisation), not just one-time output forecasting.
- Availability of time-series (or time continuous) data for monitoring, and as input to ML system.

- Domain dependence (or Domain specific services): According to the domain, a DT may provide or prioritise services specific to the industry. These are the same 'domain specific' services which exist in the physical asset (for example the optimisation problem).
- Knowledge Database: it provides the Digital Twin with the Knowledge base required to provide Services. In Order to filter out the specific Knowledge from the huge amounts of Data collected on the Internet – that is to say Big Data (BD) – these Amounts of Data must be analyzed accordingly [38].

Table 3.3 shows the characteristics presented in [39].

#### 1.4.2 Dynamic Characteristics

By leveraging these dynamic properties [28], it is possible to establish a hierarchy of digital twins.

- **Autonomy:** A digital twin can exhibit different degrees of autonomy. It can either autonomously make changes to the corresponding physical asset or allow a human operator to make modifications to the digital twin. This classification extends to various components within the twin, such as certain parts of the machine learning system or the decision-making system. Consequently, the property of autonomy can be classified as fully autonomous, non-autonomous, or partially autonomous. This classification also encompasses the self-evolution mechanism of the digital twin, specifying which changes can be made autonomously and which require human approval.
- **Synchronisation:** SThe synchronization of data in a digital twin can occur continuously or at specific time intervals. This aspect depends on factors like technology, available resources, data requirements, and

Characteristics	Definitions
Physical Entity/ Physical Twin	The physical entity/twin exists in the external real environment.
Virtual Entity/ Virtual Twin	The virtual entity/twin that exists in the virtual environment.
Physical Environment	The environment within which the PT exists.
Virtual Environment	The environment within which the virtual entity/twin exists.
State	The measured values for all parameters corresponding to the PT, DT and its environment.
Metrology	The act of measuring the state of the physical/virtual entity/twin.
Realisation	The act of changing the state of the physical/virtual entity/twin.
Twinning	The act of synchronising the states of the physical and virtual entity/twin.
Twinning Rate	The rate at which twinning occurs.
Physical-to-Virtual Connection/ Twinning	The data connections/process of measuring the state of the physical entity/twin/environment and realising that state in the virtual entity/twin/environment.
Virtual-to-Physical Connection/ Twinning	The data connections/process of measuring the state of the virtual entity/twin/environment and realising that state in the physical entity/twin/environment.
Physical Processes	The processes within which the physical entity/twin is engaged, and/or the processes acting with or upon the physical entity/twin.
Virtual Processes	The processes within which the virtual entity/twin is engaged, and/or the processes acting with or upon the virtual entity/twin.

Table 3.3: The characteristics of the Digital Twin and their descriptions.

the type of machine learning algorithm employed. A digital twin may consist of sub-components that undergo continuous synchronization for some aspects and event-based synchronization for others. The specific synchronization approach employed can result in different hierarchical structures. This synchronisation can result in different hierarchies based on the following:

- (a) How often the data is collected?
- (b) How often the data is stored?
- (c) How often the DT is updated?

#### **1.4.3 Key Characteristics Highlighted in this Paper**

In this paper, the characteristics taken into consideration are the following:

- Real-time connection with the physical entity.
- ML analysis.
- Domain dependence.
- Knowledge database
- Availability of time-series data.
- Synchronisation.

### **1.5 An Overview on the Predecessors of Digital Twins and Their Key Differences**

The process of generating virtual representations of physical objects, facilities, or processes results in the creation of virtual models that belong to a virtual space. These models are essentially computer-generated replicas of their real-world counterparts and exist within a digital environment. In this subsection, a distinction is made between the several types of Digital Models.

These Models differ primarily in how Data Flows between an original in Physical Space and its Model in Virtual Space. As can be seen in Figure 3.2, the Organization of the Data Flow in these Models is either manual and/or automatic. These three Types of Digital Models are presented below.

### **1.5.1 Digital Model / Digital Simulation Model**

In [28], the author talked about the flow of data of a DT by mentioning that it has only manual exchange of data and that it does not showcase the real-time state of the model.

Similarly, the author of [38] defines the purpose of a Digital Simulation Model: it is to replicate a system with its dynamic internal Processes in order to obtain Knowledge that can be transferred to the original Physical System.

The Simulation is mainly realized with the Support of Computers using an experimental Digital Model. This is typically carried out spontaneously and only at certain Times. In doing so, often only those Features of the original System are modeled that are of Importance for specific Problems to be solved.

As already mentioned above, the special Feature of the Use of Digital Simulation Models is that the Data between the Physical Original System and the Simulation Model is not transferred directly (automatically) in both directions – but indirectly – and often manually.

### **1.5.2 Digital Shadow**

Digital Shadow is a saved data copy of the physical state [28], it sums all the data that is left behind every time a digital service is used, such as the Internet or a mobile phone. It is a collection of data traces put together for a specific purpose and can include measured parameters as well as historical data [40].

It has a one way data flow from physical object to the digital object [28].

In the industrial sector, digital shadows represent virtual copies that are created to interact with other people and environments. It is possible to make digital shadows of digital twins because they can capture and simplify the multitude of information that they generate.

### 1.5.3 Digital Twin

The digital twin on the other hand, has fully integrated data flow where the digital twin properly reflects the actual state of the physical object.

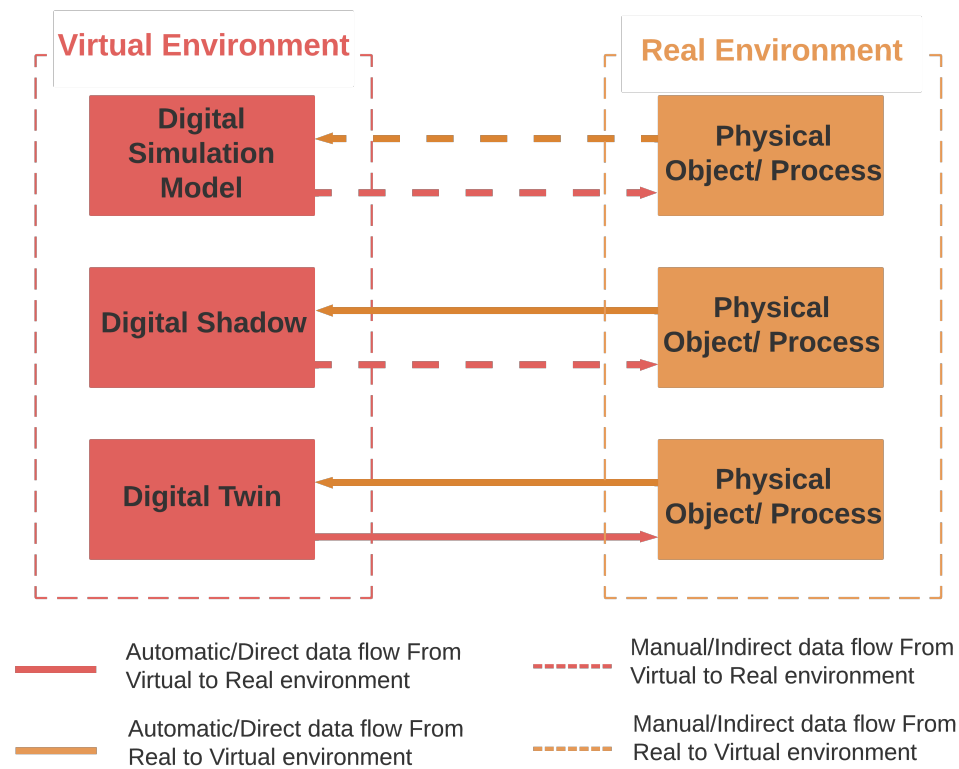


Figure 3.2: Digital Simulation VS Digital Shadow VS Digital Twin

#### **1.5.4 Digital Model VS Digital Shadow VS Digital Twin**

After defining each concept individually, a comparative analysis can be made.

Digital model, digital shadow, and digital twin are related concepts but have distinct differences.

A digital model is a computerized, data model of a building, product, or some other object that describes the form of an existing or proposed object.

A digital shadow represents virtual copies that we create to interact with other people and environments. In the industrial sector, it is used to monitor and optimize the performance of physical assets.

A digital twin is a virtual replica of a physical asset that is used to simulate, predict, and optimize the performance of the asset. It emphasizes the bi-directional approach, where the information flow not only from digital assets to the physical world but also loops back from the physical world to the digital world [41] [42] [43] [44] [45].

### **1.6 Different Types of Digital Twins**

There are four distinct types of digital twin technology [46] [47], each with its own characteristics and benefits. These types include component, asset, system, and process twins. In this subsection, each of these types are going to be seen in more detail.

To assist with visualization, Figure 3.3 is provided, which showcases an example of each type of DT.



### **1.6.1 Components Twins**

Digital models of individual components or parts, such as motors, sensors, switches, and valves, are known as component twins. These twins are the basic unit of a DT and the smallest example of a functioning component. They offer detailed information regarding a component's behavior and performance in real-time as well as over time. This enables organizations to monitor the performance and health of these components and make necessary changes whenever required.

### **1.6.2 Asset Twins**

Digital models of physical assets and when two or more components work together, such as buildings, machines, and vehicles, are referred to as asset twins. These twins provide real-time information about the operational status, performance data and environmental conditions of an asset. As a result, organizations can minimize downtime and enhance the efficiency of their operations.

### **1.6.3 System Twins/Unit Twins**

The next level of magnification involves system or unit twins, which enables to detect different assets connected to form a whole functioning system.

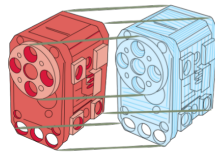
These twins facilitate the monitoring and analysis of a system's performance, helping organizations to pinpoint areas that require improvement. System twins enable organizations to optimize their processes and enhance their operational efficiency. They provide visibility regarding the interaction of assets, and may suggest performance enhancements.

#### 1.6.4 Process Twins

Digital models of entire business processes or customer journeys are referred to as process twins. It is the macro level of magnification. They furnish comprehensive information on how customers interact with an organization's products and services in real-time, assisting organizations in identifying areas where customer experience can be enhanced. They reveal how systems work together to create an entire production facility. Process twins can help determine the perfect timing schemes that ultimately influence overall effectiveness.

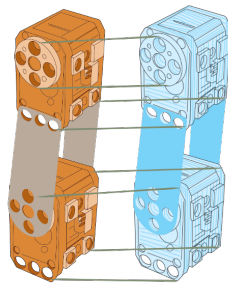
In [26], two different types of a DT have been given:

- (a) **A Dynamic DT** fed by live data flows from a physical asset, for example a building, or one of its components, like a lift motor. Insights and programmed instructions from the digital twin can then impact the physical twin using real-time control mechanisms, for example shutting down a faulty lift or adjusting the temperature of a room.
- (b) **A Static DT** that changes periodically as long-term data about a physical asset are added in. This type of digital twin is used for strategic planning, and feedback into the physical twin is achieved through the capital investment process.



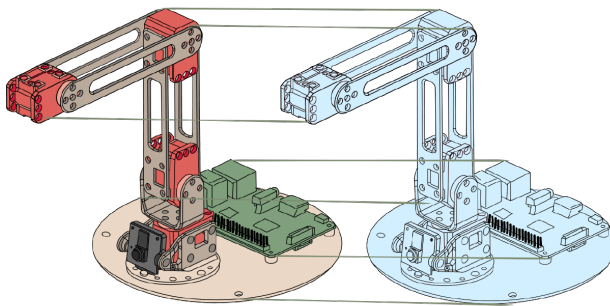
**Component Twin**

Component twins are the basic unit of digital twin, the smallest example of a functioning component. A component-level digital twin of Poppy Ergo Jr could be used to simulate the behavior of a single joint or motor in the arm. For example, the digital twin could be used to predict the torque required to move the joint, or to simulate the impact of different loads on the motor.



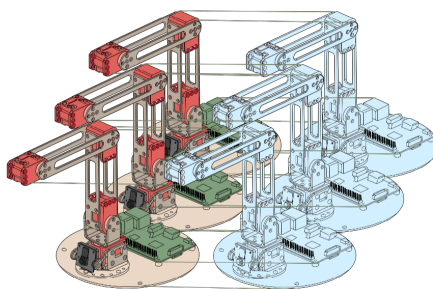
**Asset Twin**

When two or more components work together, they form what is known as an asset. An asset-level digital twin of Poppy Ergo Jr could be used to monitor the overall health of the robot arm. For example, the digital twin could collect data on the number of times the arm has been used, the types of tasks it has performed, and any maintenance or repair issues that have arisen. This information could be used to optimize the arm's performance and reduce downtime.



**System Twin**

The next level of magnification involves system or unit twins, which enable you to see how different assets come together to form an entire functioning system. A system-level digital twin of Poppy Ergo Jr could be used to simulate the behavior of the entire robotic arm system. For example, the digital twin could be used to predict the performance of the arm when performing a specific task, such as picking up an object and moving it to a new location. This could help optimize the arm's behavior and reduce the time required to complete the task.



**Process Twin**

Process twins, the macro level of magnification, reveal how systems work together to create an entire production facility. Are those systems all synchronized to operate at peak efficiency, or will delays in one system affect others? Process twins can help determine the precise timing schemes that ultimately influence overall effectiveness.

Figure 3.3: Digital Twin's types Example Robot Poppy Ergo Jr

## 2 Digital Twins with ML and DL

### 2.1 Introduction

ML and DL are an important aspect of DT technology, as they can be used to predict and analyze data in order to improve decision-making and optimize performance. There are several studies that explore the integration of ML and DL in DT technology, including the use of DL for decision support [48].

It is used in DT as well to create smart machines and plants whereby the inputs from sensors are analyzed in real-time. DT integrate IoT, AI and ML with software analytics to create digital living. The purpose of integrating DL and DT is to improve the accuracy of the DT model and to reduce the time and cost of the modeling process. ML provides important real-time insights that enhance situational awareness and enable fast, effective responses. It often can predict the future behavior of the system and provide recommendations for optimizing the system's performance [49] [50] [51].

### 2.2 ML Appliance in Digital Twins

As it is mentioned in [51], there are two widely used Data Science (specifically ML) areas used in DT that are explained down bellow and has been summarized in Table 3.4.

#### 2.2.1 Diagnostic and Predictive Analytics:

The field of IoT has brought about significant advancements in the realm of smart machines and plants. With the ability to connect a vast network of devices, IoT enables the seamless exchange of data and information between interconnected devices, systems, and humans.

As stated in [51], by integrating ML algorithms with IoT, intelligent systems that analyze and understand vast amounts of data in real-time can be created. These systems can then use this data to diagnose potential problems and predict future behaviors of the system.

The Twin is one such intelligent system that uses IoT and ML algorithms to analyze and understand inputs from various sensors in real-time. The Twin is essentially a virtual replica of the physical system, and it continually updates itself based on the data received from the sensors.

Using advanced ML algorithms, the Twin can learn from historical data and use this information to make predictions about the future behavior of the system. This ability to predict future behaviors can help prevent failures and other problems before they occur, saving time, money, and potentially even lives.

The Twin can also diagnose the causes of problems by analyzing sensor data in real-time. By identifying patterns and anomalies in the data, the Twin can quickly determine the root cause of the issue and suggest potential solutions.

In summary, IoT-based ML models, such as the Twin, are revolutionizing the way a complex system is designed and maintained.

By enabling real-time analysis and understanding of sensor data, these models can help prevent problems before they occur, improving efficiency and reducing downtime.

### **2.2.2 Prescriptive Analytics:**

Prescriptive Analytics is a field of data science that involves using advanced mathematical and computational techniques to identify optimal or feasible solutions to complex problems. Specifically, prescriptive analytics involves simulating an entire network of interconnected systems to identify the best

possible solution from a very large set of candidate solutions, given a set of variables and constraints that must be adhered to.

The primary objective of prescriptive analytics is to maximize stated business goals, such as throughput, utilization, output, and other key performance indicators. This can involve creating schedules for resources such as vehicles, personnel, and machines, to ensure maximum efficiency and productivity.

In practice, prescriptive analytics is widely used in supply chain planning and scheduling. For example, a logistics provider might use prescriptive analytics to create a schedule for its resources to ensure on-time delivery, while a manufacturer might use the technique to optimize the utilization of machines and operators to achieve maximum on-time, in-full deliveries.

To solve these complex decision-driven problems, prescriptive analytics relies on a technique called Constrained Mathematical Optimization. This involves formulating mathematical models that take into account all of the variables and constraints that must be considered in order to arrive at an optimal or feasible solution.

Powerful solvers are then used to solve these complex mathematical models, often involving millions of variables and constraints, to arrive at the best possible solution. This approach is highly effective at solving complex problems that would be too difficult or time-consuming to solve manually, and can help organizations make better decisions and achieve their stated business goals more efficiently.

To summarize, ML models predict likely outcomes for a given set of input features based on history, and Optimization models help you decide that should a predicted outcome(s) happen.

☞ **Note:** This study focuses on the first point explained in Section 2.2.1.

Diagnostic and Predictive Analytics	Prescriptive Analytics
Given a range of inputs, the Twin should be able to diagnose the causes or predict the future behavior of the system. IoT based machine learning models is what is used to create smart machines and plants whereby the inputs from sensors are analyzed in real time to diagnose, predict and thereby prevent future problems and failures before they occur.	This is where an entire network is simulated to identify an optimal or feasible solution from a very large set of candidates, given a set of variables and constraints to be adhered to, usually with the objective of maximizing stated business goals.

Table 3.4: Diagnostic and Predictive Analytics VS Prescriptive Analytics

## 2.3 Selecting an Adapted Model for IoT tabular Data

When selecting an adapted model for IoT tabular data, there are several factors to take into consideration. A few of those key considerations are mentioned down below:

- **Data Type:**

IoT devices generate various different types of data, among them, structured data, unstructured data, time-series data, etc. The chosen model should be capable of handling the specific type of data generated by the IoT devices so that it can give a good result.

- **Complexity:**

IoT data can be complex and difficult to analyze so the selected model should be able to handle the complexity of the data and provide accurate results.

- **Scale:**

IoT devices generate a large volume of data, often in real-time. The model that would be chosen should be capable of processing large amounts of data quickly and efficiently.

- **Security:**

IoT data can be sensitive and confidential. The model must have robust security features to protect the data from unauthorized access.

- **Integration:**

The model must be compatible with the existing technology stack and able to integrate with other systems and applications in the targeted organization.

- **Deployment:**

The deployment options for the model should be taking into consideration, including cloud-based, on-premises, or hybrid solutions, depending on the organization's needs.

Some popular models for IoT data analysis include ML algorithms, DL Neural Network (NN) and statistical models. It's important to evaluate different models and their capabilities as it has been done in Chapter 1, Section 3.3.

Before selecting the model that suits best the IoT data, few points needs to be specified especially concerning the data type since it would be the input of the future model. Down below are the characteristics of the selected use case that would be presented in the engineering report:

- The datatype is time series data.
- the Machine learning problem is a classification problem.
- The selected model needs to handle scalability and efficiency, since as mentioned previously, DTs can handle huge data coming from different data sources and large datasets with high-dimensional features efficiently.
- Real-time prediction.

The models that suits more these descriptions are:



- Recurrent Neural Networks.
- Decision Trees.

An article that used RNN is presented in the next subsection.

### 2.3.1 Design and development of RNN anomaly detection model for IoT networks

#### 2.3.1.1 Description

The contributions of the mentioned paper [24] is to:

- Design of an anomaly detection model for IoT networks using a RNN.
- Design of an anomaly detection model for IoT networks using CNN and RNN.
- A lightweight anomaly detection model for IoT networks using a RNN.
- Performance improvements of multiclass and binary classification models.

The focus is established on the proposed model. But first, the stages of an LSTM are viewed in details.

#### List of symbols:

$x_t$  : *Input.*

$h_t$  : *New hidden state.*

$h_{t-1}$  : *Previous hidden state.*

$C_{t-1}$  : *Previous cell state.*

$\tilde{C}_t$  : *Current cell state (Candidate).*

$C_t$  : *New cell state.*

$f_t$  : *Forget gate.*

$i_t$  : *Input gate.*

$(x)$  : *Sigmoid function.*

$\tanh(x)$  : *Tanh function.*

$W_x$  : *Gate weight.*

$b_x$  : *Gate biases.*

- (a) **Phase 1:** The initial stage of the procedure involves the implementation of the forget gate, where the determination is made regarding the relevance of specific segments within the cell state. In other words, the focus of this step is on identifying the information that should be disregarded from the cell state.

This assessment is based on the combination of the preceding hidden state and the fresh input data. And the mentioned determination is carried out by a sigmoid layer referred to as the "forget gate layer".

Through the utilization of the sigmoid activation shown in the left side of Figure 3.4, the network analyzes the values in  $h_{t-1}$  (previous hidden state) and  $x_t$  (new input data) to produce a vector where each element falls within the range of  $[0, 1]$  in the cell state  $C_{t-1}$ . A value of 1 indicates complete retention, while a value of 0 signifies complete discarding [24] [52].

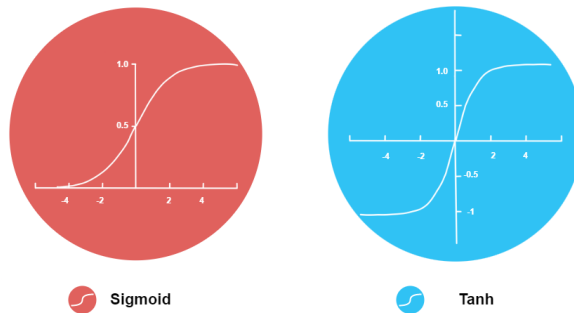


Figure 3.4: Sigmoid and Tanh Functions

The operation of the forget gate layer, which is depicted in Figure 3.5, is captured by Equation (3.1).

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (3.1)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (3.2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (3.3)$$

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t \quad (3.4)$$

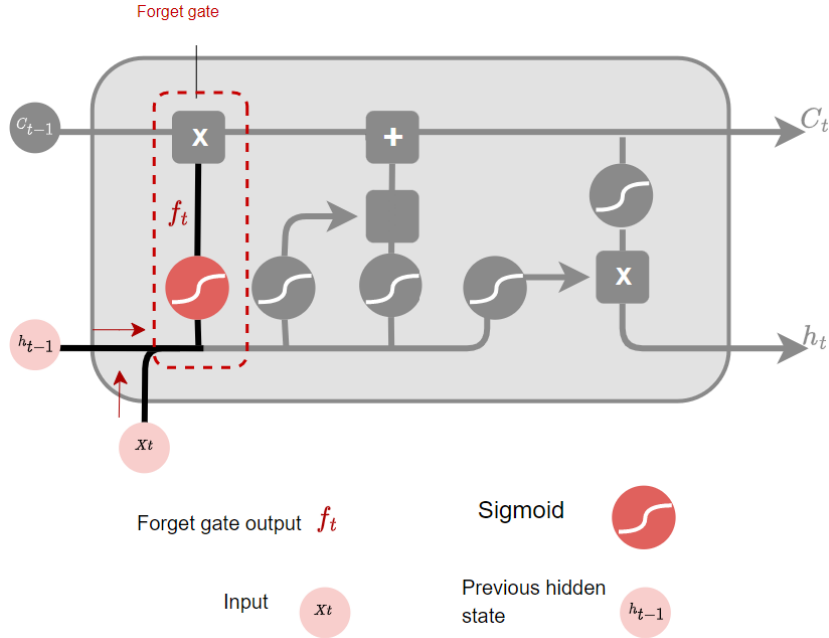


Figure 3.5: LSTM Forget Layer Operation

After generating the output values, they are multiplied element-wise with the previous cell state. This pointwise multiplication serves to diminish the impact of the cell state components that are considered irrelevant by the forget gate network. Those components receive a value close to 0, resulting in reduced influence on subsequent steps [52].

In summary, the forget gate determines which aspects of the long-term memory should be disregarded (given less weight) based on the prior hidden state and the latest data point in the sequence.

(b) **Phase 2:**

In this next step, the memory network and input gate come into play.

The objective of this stage is to identify the pertinent information to be incorporated into the long-term memory (cell state) of the network, considering the preceding hidden state ( $h_{t-1}$ ) and the fresh input data ( $x_t$ ).

The New Memory Network:

it is a tanh activated neural network which has learned how to combine the previous hidden state and new input data to generate a ‘new memory update vector’. This vector essentially contains information from the new input data given the context from the previous hidden state. This vector tells us how much to update each component of the long-term memory (cell state) of the network given the new data [52].

The tanh function has been used in this context because its output values range from -1 to 1, allowing for the inclusion of negative values. The inclusion of negative values is crucial for the intent of diminishing the influence of a component in the cell state.

Input Gate:

In the first part mentioned above, which involves generating the new memory vector, a significant issue arises. It fails to assess whether the new input data holds any significance worth remembering. This is where the input gate comes in.

The input gate operates as a filter, employing a sigmoid-activated network to identify the components of the "new memory vector" that are worth retaining. By producing a vector of values ranging from 0 to 1 (due to the sigmoid activation), the input gate functions as a filter through pointwise multiplication. Similar to our observations with the forget gate, an output value close to zero indicates that the corresponding element of the cell state should not be updated.

Output: The outputs from the first and second parts are multiplied element-wise. This operation ensures that the magnitude of the newly chosen information determined in the second part is regulated and set to 0 if necessary.

The resulting combined vector is then added to the cell state, effectively updating the network's long-term memory [52].

The operation of the Input gate layer, which is depicted in Figure 3.6, is captured by Equation (3.2) and Equation (3.3).

(c) **Phase 3:**

In LSTM networks, the cell state refers to the memory component that carries information throughout the sequence. It serves as a form of long-term memory that allows the network to retain information over longer periods, mitigating the vanishing gradient problem.

The cell state acts as an information highway, enabling the LSTM to preserve relevant information and discard irrelevant information over time. It runs parallel to the hidden state and undergoes a series of operations such as addition, multiplication, and modulation through gates (input

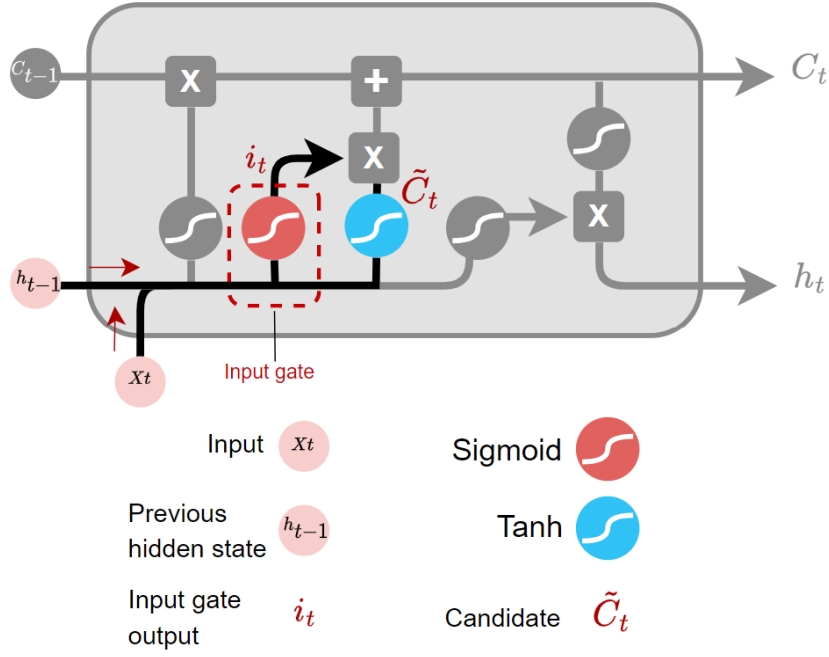


Figure 3.6: LSTM Input Gate Layer Operation

gate, forget gate, and output gate) to regulate the flow of information.

The cell state serves as the primary component that captures the network's memory and plays a crucial role in retaining and updating information throughout the sequence processing in LSTM networks. It is presented in Figure 3.7 and is captured by Equation (3.4).

(d) **Phase 4:**

In order to ensure that only essential information is outputted and saved to the new hidden state, we apply a filter to the updated cell state. However, before applying the filter, we subject the cell state to a tanh function, which confines the values within the range of  $[-1, 1]$ .

Here is the step-by-step process for this final step [52]:

- The current cell state is pointwise transformed using the tanh function, resulting in the squished cell state that now resides within the

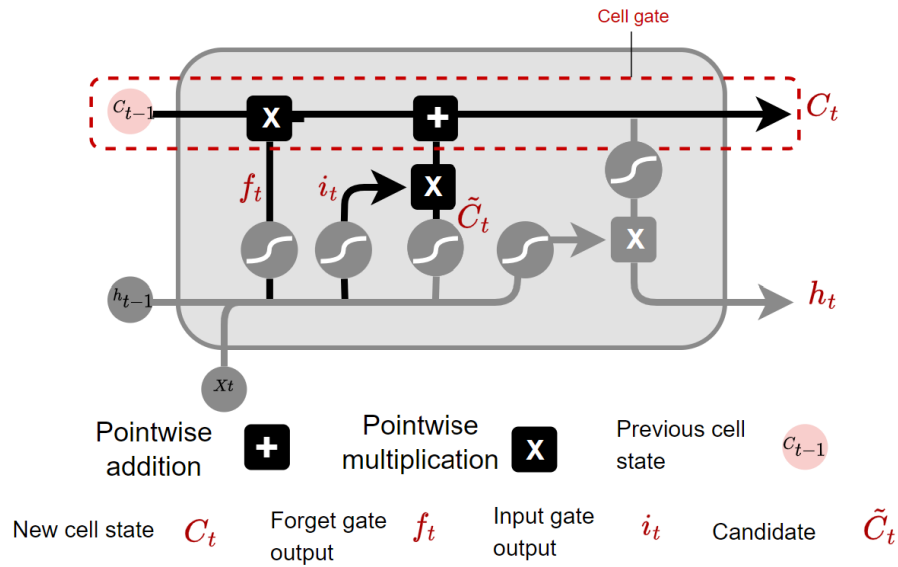


Figure 3.7: LSTM Cell State Operation

interval of  $[-1, 1]$ .

- Both the previous hidden state and the current input data are passed through a sigmoid-activated neural network, generating the filter vector.
- The squished cell state is then multiplied pointwise with the filter vector obtained from the previous step.
- The resulting output becomes the new hidden state.

This process ensures that the outputted hidden state only contains pertinent information by applying the filter derived from the sigmoid network to the transformed cell state.

This step is presented in Figure 3.8.

### Why RNNs instead of another Model?

The concerned article mentioned that DL techniques gained popularity due to their ability to detect computer network threats and abnormalities in various applications and that an RNN model has shown to be effective in multiple

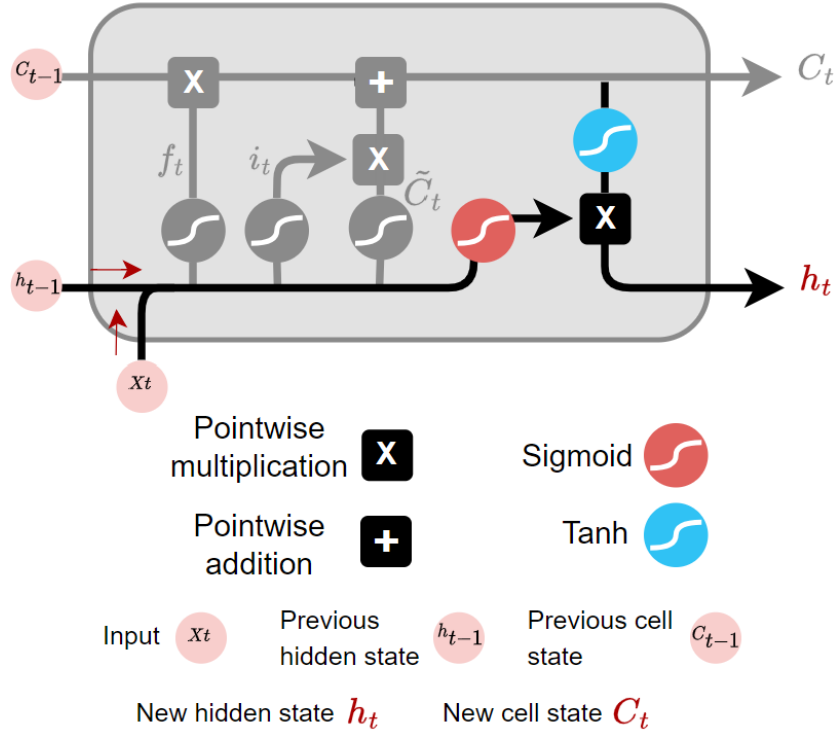


Figure 3.8: LSTM Output Gate Operation

areas due to its better capability, so their realised model consists of an input layer, output layer, and four recurrent, activation, normalization, activity regularization and dropout layers.

However, there are frequent reports and articles stating that Tree-Based Models tend to achieve superior performance compared to Neural Networks.

### 2.3.2 Why do tree-based models still outperform deep learning on tabular data?

In this article [53], 45 tabular datasets has been used to perform a comparison between various models. Those datasets has been selected depending on different characteristics and differs on:

- Heterogeneous data.
- Real-world data.



- Not deterministic.

The selected models are :

- Scikit Learn’s RandomForest.
- GradientBoostingTrees (GBTs) (or HistGradientBoostingTrees when using categorical features).
- XGBoost.
- MLP.
- Resnet.

Figure 3.9 represents the results on medium-sized datasets with only numerical features. Dotted lines correspond to the score of the default hyperparameters. Each value corresponds to the test score of the best model (on the validation set) after a specific number of random search iterations, averaged on 15 shuffles of the random search order. The ribbon corresponds to the minimum and maximum scores on these 15 shuffles.

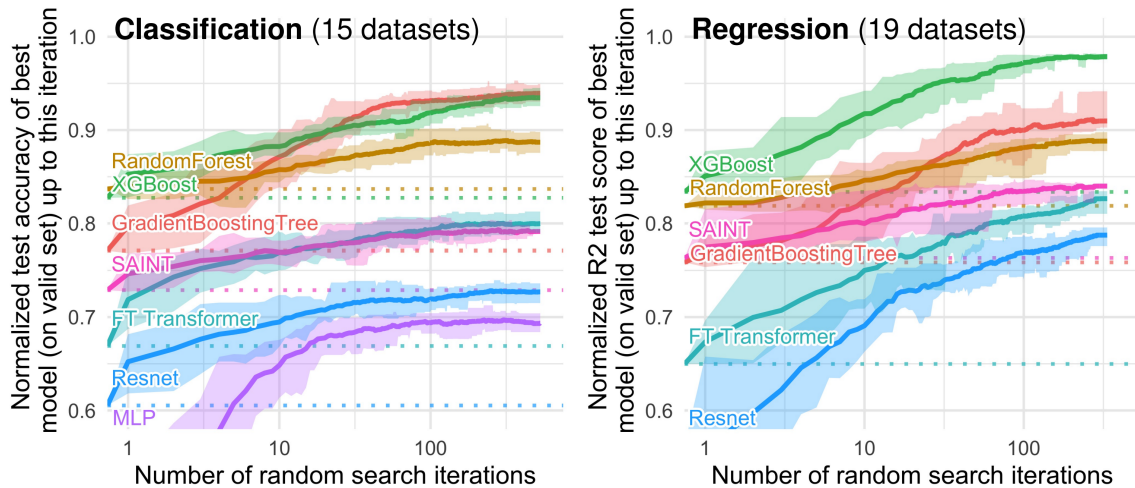


Figure 3.9: Results on medium-sized datasets with only numerical features

And Figure 3.10 represents results on medium-sized datasets, with both numerical and categorical features.

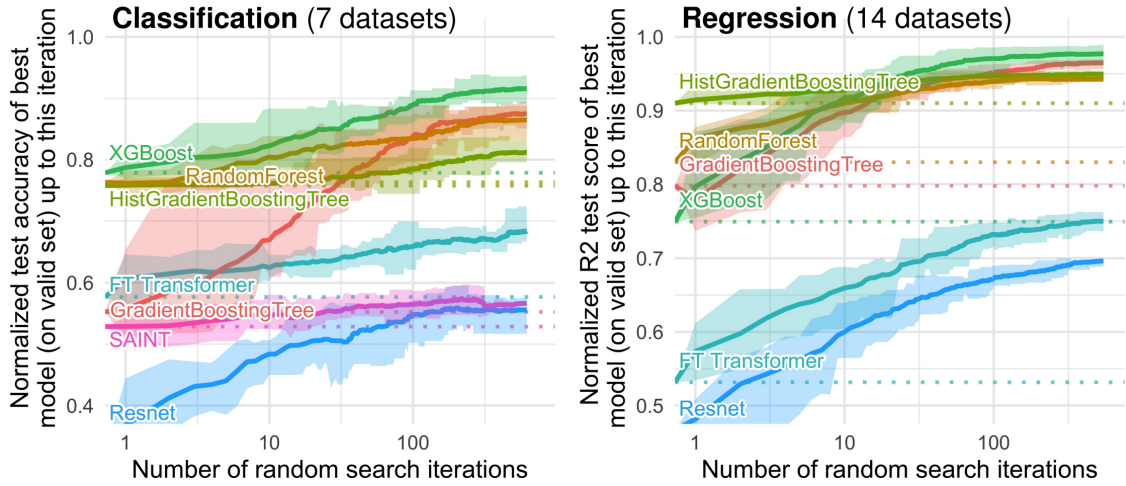


Figure 3.10: Results on medium-sized datasets, with both numerical and categorical features

And it has been proven that tuning the hyperparameters does not make the NNs perform better than tree-based model.

### 3 The use of Digital Twins for Resilience and Prevention

#### 3.1 Digital twins as run-time predictive models for the resilience of cyber-physical systems: a conceptual framework

The objective of the paper [54] is to propose a new approach for enhancing the resilience of Cyber-Physical Systems (CPSs) by using DT as run-time pre-

dictive models.

CPSs are complex systems that intricate combinations of physical components and digital technologies. They are increasingly utilized in critical domains such as transportation, healthcare, and energy systems. However, these systems are susceptible to disruptions and failures, which can result in severe consequences, including safety hazards, financial losses, and damage to reputation.

Similar to IoT systems, which serve as our study case, disturbances, anomalies, and interference pose significant challenges in handling time-series data within various Information Technology (IT) and IoT systems. Given the enormous volume of time-series data generated by multiple sensors daily, manual anomaly detection by humans is no longer feasible.

Thus, the objective of the referenced article is to develop an approach that enhances the resilience of CPSs and facilitates their adaptation and recovery from disruptions. Existing resilience strategies for CPSs primarily focus on reactive measures such as detection systems and recovery. However, these measures may not sufficiently address the growing complexity and unpredictability of CPSs.

Existing approaches to CPSs resilience often focus on reactive measures such as a detection system and recovery. However, these measures may not be sufficient to address the increasing complexity and unpredictability of CPSs. The authors argue that DT can provide a proactive, predictive approach to enhancing CPSs resilience.

They can predict potential failures and recommend actions to prevent them, thus enabling CPSs to anticipate and respond to disruptions more effectively.

Thoroughly, the objective of the paper is to propose a conceptual framework for using DT as run-time predictive models to enhance the resilience of CPSs.

The authors aim to demonstrate that this approach can significantly improve the performance, safety, and reliability of CPSs, and can reduce downtime and maintenance costs. The paper also aims to contribute to the field of CPSs by highlighting the potential of DT as a tool for enhancing resilience and providing a framework for further research into their use in CPSs applications.

To adapt this paper to our specific problematic which is enhancing the resilience of IoT systems by using DT , here is what can be extracted :

- The paper highlights the importance of resilience in the context of CPSs and argues that resilience is not just about recovering from disruptions, but also about adapting to changing conditions and mitigating the impact of disruptions.

This is particularly relevant for IoT systems, which are often subject to a wide range of potential disruptions, such as network outages, cyber-attacks, and environmental factors. By understanding the importance of resilience, An effective approach to enhancing the resilience of an IoT system can be developed.

- The paper proposes a conceptual framework for creating DT of CPSs and using them as run-time predictive models. This approach can be applied to IoT systems as well.

DT can help to predict and prevent disruptions in IoT systems, and can provide a tool for testing and optimizing these systems in a virtual environment. By considering the use of DT in an IoT system, its resilience and performance improves.

- It has been suggested to explore the potential of DT for enhancing the social and environmental sustainability of CPSs. This is equally relevant for IoT systems, where sustainability is an increasingly important con-

cern. For example, the DT may be used to optimize energy usage, reduce waste, or improve the environmental impact of the IoT system.

### **3.2 Cognitive Digital Twins for Resilience in Production: A Conceptual Framework**

Similar to [54], The objective of this article [55] is to propose a framework to enhance the resilience, but instead of CPS, it is for production systems using cognitive DT.

Why production systems? Because they have become more intricate and inter-dependent in recent times, making them susceptible to a range of disruptions and uncertainties. To tackle these challenges, experts and researchers are exploring fresh approaches to boost the resilience of production systems. One such approach involves using cognitive DT to improve the system's ability to withstand disruptions and uncertainties.

The article aims to explore the concept of cognitive DT, which are DT that incorporate AI and ML to enhance their capabilities. These DT can provide real-time feedback to operators, predict potential issues before they occur, and optimize production processes.

Here is what this paper discussed and tried to attain as objectives:

- Discuss how cognitive DT can be used to improve production processes and increase resilience. For example, DT can help identify potential issues in the production process and provide recommendations for addressing them, reducing the risk of disruptions.
- Explain how DT can help optimize the production process by simulating different scenarios and identifying the most efficient production methods.
- Elucidate how Cognitive Digital Twin (CDT) presents several challenges,

among them data privacy concerns, the need for significant computing power, and the complexity of integrating DT into existing production systems.

To give a solution to these problems and challenges, the goal is to provide a roadmap for the development and deployment of CDT in production systems. The roadmap includes several steps, such as identifying the Key Performance Indicators (KPIs) that the DT will monitor, selecting the appropriate AI and ML algorithms, and developing a data management strategy.

- Provide a framework for using CDT to enhance the resilience of production systems. By leveraging the power of AI and ML, CDT can help production systems adapt to changing conditions, reduce the risk of disruptions, and improve overall efficiency and productivity.

### **3.3 State of the Art in using Digital Twins for prevention**

- In [56], Koen Bruynseels, Filippo Santoni de Sio and Jeroen van den Hoven used Digital Twins in healthcare to reflect the current state of physical objects by redefining 'normality' and 'health' based on individual patterns compared to population patterns, impacting the distinction between therapy and enhancement. The concept of Digital Twins is a valuable tool for analyzing the ethical and conceptual aspects of future healthcare and human enhancement by utilizing individualized data on molecular makeup, physiology, lifestyle, and diet. Comparing Digital Twins across populations helps differentiate between health and disease, shaping the therapy-enhancement debate. Digital Twins have the potential to identify effective routes for therapy and enhancement, allowing individuals to define their well-being preferences. However, ethical, legal,

and social concerns arise, including challenges to equality and the risk of discrimination based on compiled information. Governance is necessary to ensure transparency, data privacy, and fair access to this data-intensive technology.

- In [57], the authors presented the benefits of using digital twins in manufacturing. Six core cognitive capabilities (perception, attention, memory, reasoning, problem-solving, and learning) were described along with their ability to influence complex manufacturing decisions and future autonomy.
- The research paper [58] presents a novel framework for anomaly detection in digital twin-based Cyber-Physical Systems (CPS). The framework includes two main components: a discrepancy detector based on the Gaussian Mixture Model (GMM), and an anomaly classifier utilizing the Hidden Markov Model (HMM).

Initially, the discrepancy detector analyzes data from two sources: one from the physical plant and the other from the digital twin. It assesses if there are any anomalies present by comparing the data from both sources. The generated signatures from this detector are then used by the anomaly classifier to classify different types of anomalies, employing the HMM.

To validate the effectiveness of the framework, experiments were conducted using the Tennessee Eastman process model.

In future endeavors, the researchers aim to enhance the framework by integrating correction mechanisms. These mechanisms would be designed to maintain system stability based on the classification results obtained from the anomaly classifier.

- In the paper [59], a pioneering approach is introduced for constructing a dynamic digital replica, or digital twin, of an additive manufacturing system utilizing retrofitted low-end sensors found in IoT devices. By

leveraging side-channels like acoustic, vibration, magnetic, and power signals, the system can be indirectly monitored. These signals are then processed using a clustering algorithm to generate a comprehensive fingerprint library that accurately represents the physical state of the system, essentially creating a physical twin in the digital realm. The digital twin serves the purpose of detecting and pinpointing anomalous physical emissions that may lead to variations in product quality.

With an average accuracy of 83.09%, the digital twin successfully localizes errors by comparing the detected emissions to the established fingerprint library. Furthermore, an algorithm is presented for updating the digital twin and deducing any deviations in quality. To illustrate the effectiveness of the methodology, a case study is conducted using an additive manufacturing system.

In comparison to existing methods that disregard the liveliness of the model, their created approach outperforms them by dynamically updating itself, accurately inferring quality deviations, and precisely localizing abnormal faults within the additive manufacturing system.

## 4 Digital Twins architecture

The objective of the article "Towards a Requirement-driven Digital Twin Architecture", as it is mentioned in its title, is to propose a new architecture for DT that is driven by requirements. Since DT can be used to simulate, predict, and optimize the behavior of the physical systems in real-time, its development and realization of an architecture independently of the use case can be challenging especially due to the need for accurate data, modeling, and simulation.

To address these challenges, the authors propose a requirement-driven ap-



proach to the design of DT architectures. This approach emphasizes the importance of understanding and defining the requirements of the physical system before developing the DT. The authors suggest that a set of requirements can serve as the basis for the DT architecture, and that this architecture can be designed to meet these requirements.

The provided comprehensive and practical approach to the development of DT architectures that can be used to support a range of applications and industries will be presented in the next subsection.

As mentioned, The paper proposes a conceptual framework for the development and deployment of CDT in production systems. This framework provides a systematic approach for integrating DT technology into production systems and can help practitioners and researchers to implement DT in a systematic and effective manner.

This conceptual framework is presented in Figure 3.11.

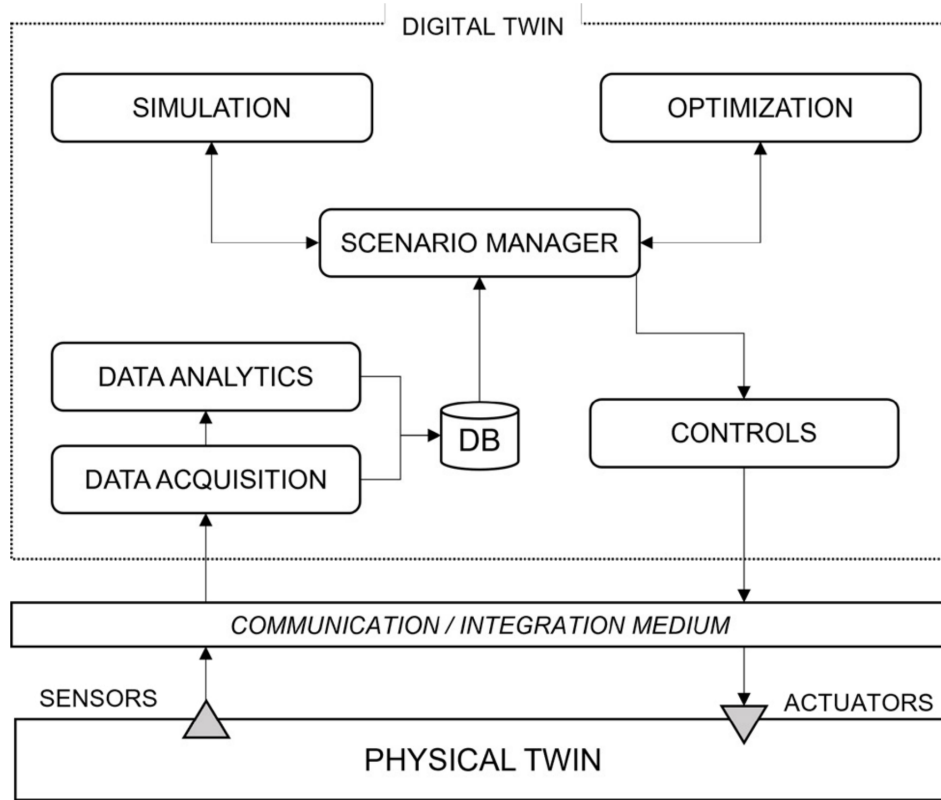


Figure 3.11: Exemplary DT architecture.

## 5 Framework CSDT

Taking into consideration the following mentioned points:

- Level of granularity: System Twin.
- Use AI for Diagnostic and Predictive Analytics.
- The creation of a static DT.

By remaining faithful faithful to the above points, a CSDT can be created, this term that is created to represent the framework that can not only replicate the PT actions, what makes it a super-DT is its ability to generate disturbed data and functions to make it more resilient to future problems rather waiting for it to occur.

## 5.1 Conclusion

In conclusion, this state-of-the-art analysis and different reviews on the present articles has shed light on the power Digital Twins and Cognitive Digital Twins. Through an exploration of definitions, current research, and practical applications, it is evident that Digital Twins offer significant benefits for resilience in various domains.

Further experimental work will be emphasizing on :

- Create an Edge/Fog System of Systems Architecture adapted to the problem and use-case presented.
- Use RNN for a classification problem.
- Prove that Decision Trees perform better on tabular data.
- Create a Digital Twin adapted perfectly to the physical twin.
- Generate perturbations and anomalies to make it a super-DT.

Overall, this state-of-the-art analysis highlights the need for continued research, innovation, and investment in Digital Twin technologies. With further advancements and integration into real-world applications, Digital Twins have the potential to revolutionize maintenance and resilience practices, leading to improved operational efficiency, reduced downtime, and increased system performance.

## Part IV

### Contribution

# Chapter 4

## Design and Implementation

### Contents

---

1	Introduction . . . . .	<b>83</b>
2	Use Case . . . . .	<b>83</b>
2.1	Description of the Use Case . . . . .	83
2.2	Description of the used Dataset . . . . .	84
3	Used Technologies and Hardware . . . . .	<b>85</b>
3.1	Raspberry Pi 3 Model B . . . . .	85
3.2	GrovePi+ . . . . .	87
3.3	Grove Sensors . . . . .	87
3.4	RabbitMQ - MQTT . . . . .	91
3.5	InfluxDB . . . . .	93
3.6	Poppy Ergo Jr . . . . .	93
3.7	Computer . . . . .	94
4	General Architecture . . . . .	<b>94</b>
4.1	Simplified Architecture of the System . . . . .	94
4.2	Detailed Architecture of the System . . . . .	97
4.3	Elaborated Architecture of the System . . . . .	101
5	Implementation . . . . .	<b>108</b>

5.1	Class Diagram . . . . .	108
5.2	Creating a Dataset . . . . .	120
5.3	Selecting an ML or DL Models . . . . .	126
5.4	Languages and Libraries . . . . .	148

---

# 1 Introduction

After leveraging and exploiting the state of the art that defined a Digital Twin's essential concepts, advanced methods and frameworks that used DTs for detecting anomalies and disturbance, and the selection of an adapted model going through both traditional machine learning algorithms and deep neural networks. The main aim and objective of this study is to make a considerable contribution in the field of Digital Twins by developing, implementing and testing a Cognitive Digital Twin of an IoT system that not only is the perfect replica of the Physical Twin but also generate disturbance to be dealt with later on.

This chapter will be organized as follow:

- Describing the use case.
- Present the used technologies and hardware.
- Exhibiting the general architecture.
- Displaying and explaining the implementation.

## 2 Use Case

The use case followed to accomplish the wanted system is the following:

### 2.1 Description of the Use Case

The use case is Having an IoT system that corresponds to the Physical Twin and that has four sensors:

- (a) A temperature sensor.

- (b) A humidity sensor.
- (c) A light sensor.
- (d) A CO2 sensor.

Each sensor corresponds to a feature in the **Occupancy Detection Data Set** taken from the website University of California, Irvine (UCI) Machine Learning Repository<sup>1</sup>. The dataset is described in subsection 2.2.

At first hand, an adapted model is trained on that dataset so that the resulted prediction would lead to future actions. The selection and training of the adapted model is explained in subsection 5.3.

The Digital Twin would be the perfect replica of this Physical Twin but what makes it a super-Digital Twin is the capacity of detecting disturbances.

## 2.2 Description of the used Dataset

The Occupancy Detection Data Set<sup>2</sup> is an experimental data used for binary classification (room occupancy) from Temperature, Humidity, Light and CO2.

Date time is given in the following form: year-month-day hour:minute:second  
 Temperature is in Celsius (°C) Relative Humidity is in percentage (%) Light is in Lux CO2 is in ppm Humidity Ratio, Derived quantity from temperature and relative humidity, in kgwater-vapor/kg-air Occupancy, 0 or 1, 0 for not occupied, 1 for occupied status.

Mode details are given in Table 4.1.

---

<sup>1</sup>UCI Machine Learning Repository is a widely-used online collection of datasets for machine learning research, maintained by the University of California, Irvine. It provides researchers and practitioners with access to diverse datasets that have been preprocessed and formatted for use in machine learning experiments. The repository promotes reproducibility, fair comparisons, and knowledge sharing in the field of machine learning.

<sup>2</sup>**Occupancy Detection Data Set:** <https://archive.ics.uci.edu/ml/datasets/Occupancy+Detection+>



<b>Dataset Characteristics</b>	Time-Series	<b>Number of Instances</b>	20560
<b>Attribute Characteristics</b>	Real	<b>Number of Attributes</b>	7
<b>Associated Tasks</b>	Classification	<b>Missing Values</b>	N/A

Table 4.1: The Description of the External Dataset (Occupancy Dataset).

### 3 Used Technologies and Hardware

Since it is a decentralised IoT system, different hardware and technologies used for this system are presented in the upcoming subsections which are:

- Two Raspberry Pi 3 Model B.
- GrovePi+ add-on board.
- Grove sensors.
- Poppy Ergo Jr and its OS.
- RabbitMQ.
- MQTT.
- InfluxDB.
- Computer.

#### 3.1 Raspberry Pi 3 Model B

The generation of Raspberry Pi used on this project is the Raspberry Pi 3 Model B. It is a single-board computer developed by the Raspberry Pi Foundation. It is the third generation of the Raspberry Pi series, succeeding the Raspberry Pi 2 Model B and the final revision of this third generation [60]. It is presented in Figure 4.1

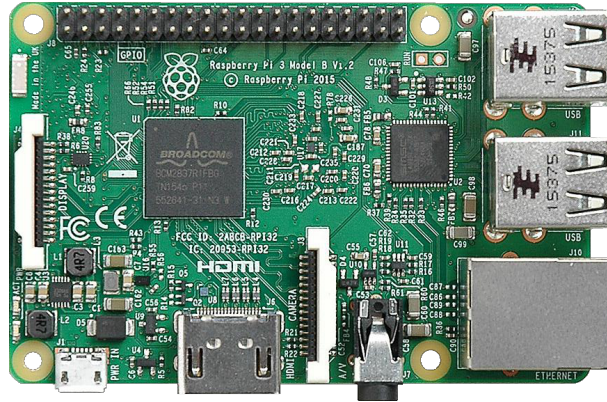


Figure 4.1: Raspberry Pi 3 Model B

It has improved and has many new features compared to its predecessors. A few of its notable The Raspberry Pi 3 Model B features several key components and improvements compared to its predecessors. Few of its notable features and specifications are mentioned in the subsubsection 3.1.1.

### 3.1.1 Features and Specifications

- (a) **Processor:** it has a quad-core 64-bit ARM Cortex-A53 CPU running at 1.2 GHz.
- (b) **Memory:** It has 1 GB of LPDDR2 RAM.
- (c) **Connectivity:** The board includes built-in Wi-Fi 802.11n and Bluetooth 4.
- (d) **USB and Ethernet:** it is composed of four USB 2.0 ports and a 10/100 Ethernet port.
- (e) **Video and Audio:** it supports full HD (1080p) video playback and includes an HDMI port for connecting to displays or TVs. It also features a 3.5mm audio jack for audio output.

### **3.1.2 Chosen OS**

The chosen OS that has been flashed on a 16 GB SD Card is Raspbian "Raspberry Pi OS with desktop and recommended software (32 bits)".

It is no surprise that Raspbian tops this list due to its importance to the Raspberry community. Raspbian is an independent distro built for Raspberry Pi 3. Its popularity is because it is one of the oldest operating systems to be used with the earlier versions of the Raspberry Pi.

Raspbian is the chosen OS due to its importance to the Raspberry community. It is an independent distribution built for Raspberries. Its popularity is attributable to it being the oldest OS and that it was being used with earlier versions of the Raspberry Pi.

In terms of build, Raspbian is based on Debian Linux and it comes with approximately 35,000 packages in a bundle compatible with Raspberry Pi 3 [61].

## **3.2 GrovePi+**

GrovePi+ is add-on board with 15 Grove 4-pin interfaces that brings Grove sensors to the Raspberry Pi. It is very convenient to bring and connect various Grove modules with a simple plug-and-play functionality [62].

It provides digital, analog, and I2C interfaces and is presented in Figure 4.2

## **3.3 Grove Sensors**

The used grove sensors for this specific use case are:

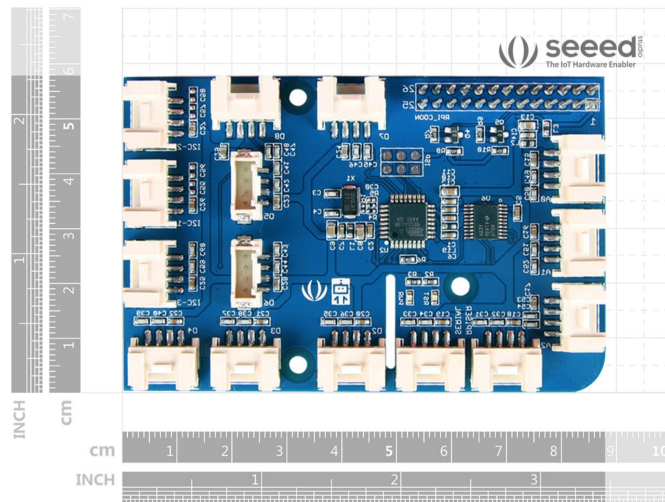


Figure 4.2: GrovePi+.

- Grove Temperature&Humidity Sensor (DHT11).
- Grove - Barometer (High-Accuracy).
- Grove - Light Sensor.
- Grove-VOC and eCO2 Gas Sensor(SGP30).

### 3.3.1 Grove Temperature&Humidity Sensor (DHT11)

The Temperature&Humidity sensor [63] provides a pre-calibrated digital output. It has the following features:

- Relative Humidity and temperature measurement.
- Full range temperature compensation Calibrated.
- Digital signal.
- Long term stability.
- Long transmission distance(>20m).
- Low power consumption.

It is represented in Figure 4.3.

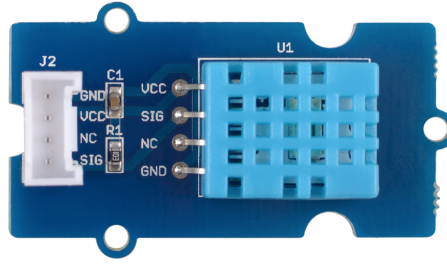


Figure 4.3: Grove DHT11.

### 3.3.2 Grove - Barometer (High-Accuracy)

This Grove - Barometer (High-Accuracy) Sensor [64] features an HP206C high-accuracy chip to detect barometric pressure, Altimeter, and temperature. It can widely measure pressure ranging from 300mbar 1200mbar, with super high accuracy of 0.01mbar(0.1m) in ultra-high resolution mode.

Its features are:

- Digital two wire (I2C) interface.
- Programmable Events and Interrupt Controls.
- Wide barometric pressure range.
- Flexible supply voltage range.
- Ultra-low power consumption.
- Altitude Resolution down to 0.01 meter.
- Temperature measurement included.

It is represented in Figure 4.4.

### 3.3.3 Grove - Light Sensor

The Grove - Light sensor integrates a photo-resistor(light dependent resistor) to detect the intensity of light [65]. The resistance of photo-resistor decreases

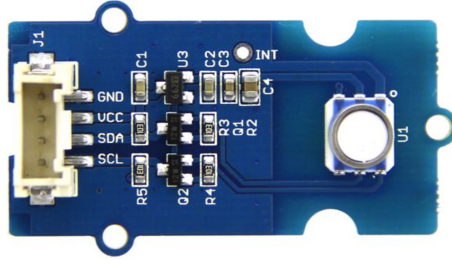


Figure 4.4: Grove - Barometer Sensor(High-Accuracy).

when the intensity of light increases. A dual OpAmp chip LM358 on board produces voltage corresponding to intensity of light(i.e. based on resistance value). The output signal is analog value, the brighter the light is, the larger the value.

It is represented in Figure 4.5.

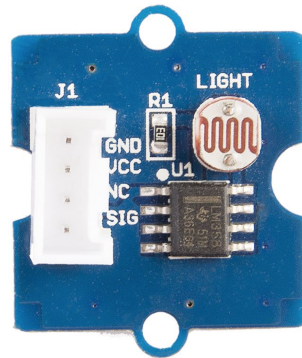


Figure 4.5: Grove - Light Sensor.

### 3.3.4 Grove-VOC and eCO2 Gas Sensor(SGP30)

The Grove-VOC and eCO2 Gas Sensor(SGP30) is an air quality detection sensor [66]. This grove module is based on SGP30, we provide TVOC(Total Volatile Organic Compounds) and CO2eq output for this module.

The SGP30 is a digital multi-pixel gas sensor designed for easy integration into air purifier, demand-controlled ventilation, and IoT applications.

It is represented in Figure 4.6.

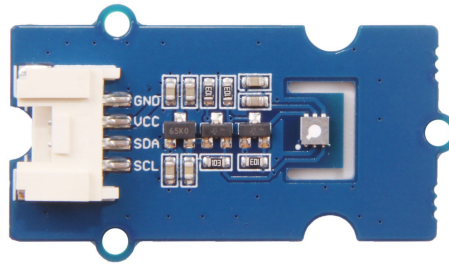


Figure 4.6: Grove-VOC and eCO2 Gas Sensor(SGP30).

### 3.4 RabbitMQ - MQTT

#### 3.4.1 RabbitMQ

RabbitMQ is an open-source message-broker software that provides a messaging queue model for exchanging messages between different applications or components within a distributed system [67]. It is built on the Advanced Message Queuing Protocol (AMQP) standard, which allows applications to communicate and transfer data reliably, asynchronously, and in a loosely coupled manner.

RabbitMQ features are:

- Asynchronous Messaging.
- Distributed Deployment.
- Management and Monitoring.

#### 3.4.2 RabbitMQ MQTT

RabbitMQ MQTT is an extension to RabbitMQ that enables support for the Message Queuing Telemetry Transport (MQTT) protocol [68]. MQTT is a

lightweight messaging protocol designed for efficient communication between devices or client applications in constrained or unreliable networks.

RabbitMQ MQTT allows devices or clients to connect to RabbitMQ as an MQTT broker and exchange messages using the MQTT protocol. It provides seamless integration between MQTT clients and the RabbitMQ messaging system, enabling interoperability with other messaging protocols and systems supported by RabbitMQ.

RabbitMQ MQTT extends the capabilities of RabbitMQ by adding MQTT-specific features such as support for MQTT Quality of Service levels, retain messages, last-will-and-testament messages, and session persistence for MQTT clients.

By combining RabbitMQ's robust messaging infrastructure with the MQTT protocol, RabbitMQ MQTT provides a scalable and flexible solution for building IoT applications and other messaging systems that require lightweight, efficient, and reliable communication.

### **3.4.3 Why RabbitMQ - MQTT and not another Message broker - protocol ?**

There are several reasons why RabbitMQ MQTT may be a suitable choice for an IoT application among them:

- MQTT is lightweight, efficient and designed specifically for constrained devices and unreliable networks.
- RabbitMQ is known for its scalability and ability to handle large numbers of concurrent connections and messages.
- RabbitMQ MQTT integrates seamlessly with other messaging protocols supported by RabbitMQ.



- RabbitMQ MQTT leverages RabbitMQ’s robust messaging infrastructure, providing reliable message delivery, message persistence, and fault-tolerance. It ensures that messages are not lost even in the case of network or device failures.
- It supports topic-based message routing, allowing devices to subscribe to specific topics of interest and receive relevant messages. Additionally, RabbitMQ provides features like message filtering and transformation, enabling data processing and transformation within the messaging system itself.

### 3.5 InfluxDB

InfluxDB is an open-source time series database developed by the company InfluxData[69]. It is written in the Go programming language for storage and retrieval of time series data in fields such as operations monitoring, application metrics, Internet of Things sensor data, and real-time analytics.

### 3.6 Poppy Ergo Jr

Ergo Jr is a low cost arm designed for education, easy to build and modify[70][71].

It is a robotic arm, consisting of 6 motors allowing life-like movements and 3D printed elements. The use of rivets make the assembly, modification and reassembly easy. Ergo Jr comes with three tools for different interactions with its environment:

- A lampshade.
- A gripper.
- A pen holder.

The one used in our use case is a Gripper.

The robot is controlled with a Raspberry Pi board, and a camera helps it interact with the world (which is not used in this use case).

### **3.7 Computer**

The used computer is a DELL XPS 13 9305 running on Windows 10 and has these specifications:

- Precessor: 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 2.80 GHz
- RAM: 16GO.
- System Type: 64-bit operating system, x64-based processor.

## **4 General Architecture**

In this section, the general architecture of the realised system is presented in the form of three levels of complexity:

- The system's architecture in a simplified level.
- The system's architecture in a detailed level.
- The system's architecture in an elaborate level.

The architecture follows the structure given in [72] and the characteristics given in [28] [37] [38] [39].

### **4.1 Simplified Architecture of the System**

This subsection permits to view the architecture from an overarching standpoint, allowing to evaluate the system from a comprehensive or broad perspective. It implies looking at the larger picture or taking into account the overall

view or perspective rather than focusing on specific details. It suggests considering the subject matter or situation in a more holistic or all-encompassing manner.

Figure 4.7 represents the simplified form of the architecture.

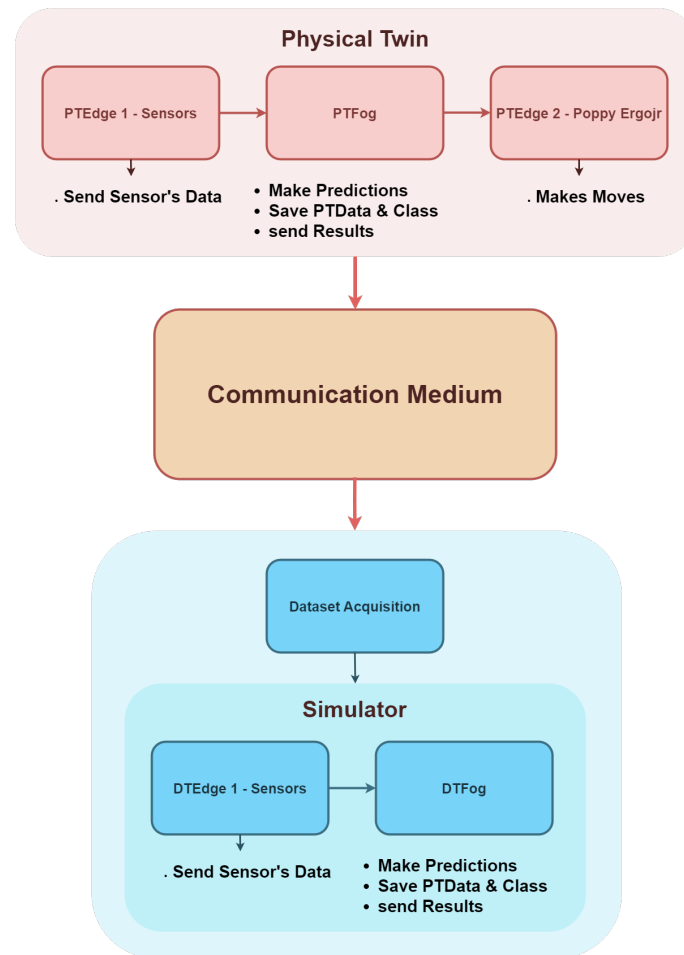


Figure 4.7: The system’s architecture in a simplified form.

#### 4.1.1 Description of the Simplified Architecture

##### 4.1.1.1 Physical Twin

The physical twin is in the form of a decentralised system.

- **The First sub-system:** This first sub-system is in the form of an Edge/Fog architecture due to Bandwidth efficiency, low-latency processing and storage.
  - PTEdge 1 - Sensors: This edge represents mainly the sensors. It collects the sensor's data and sends it to its fog for processing.
  - PTFog: The corresponding fog accomplishes several functions among them:
    - \* Train a model using the dataset mentioned in subsection 2.2.
    - \* use that trained model to make predictions on the received PTEdge 1 Data.
    - \* Save the Sensor's Data for future periodic trainings.
    - \* Send the collected data and results to both communication medium as well as the second Edge.
- **PTEdge 2 - Poppy Ergo Jr:** Poppy Ergo Jr has its own Operating System that is why it is in a decentralised system. This robot and its Operating System are better explained in subsection 3.6. The actions and moves performed by the robot depends on the data and results it received from the fog of the first sub-system.

#### 4.1.1.2 Communication Medium

The medium serves as a tunnel, a channel or a pathway, through which every interaction and communication occurring between the PT and its associated DT is transported and securely stored. This ensures that all the data and information passing through and exchanged between the two twins are seamlessly transmitted for further analysis, monitoring and synchronization.

#### 4.1.1.3 Digital Twin

The DT, the replica of the PT has different essential components, inspired from [72] among them:

- **Dataset Acquisition Module:** since the origin of the data that the DT processes comes from different sources, this module serves to collect the data and process it to combine it and make it one global source of data or dataset.
- **Simulator:** it is the core of the proposed DT which is a faithful replication of the PT's functionality. But in this study, only the first sub-system is being replicated. Additionally, the simulator generates and address certain types of disturbances to offer additional insights to support decision-making, predicting anomalies or future failures.

## 4.2 Detailed Architecture of the System

This subsection permits to view the architecture in an in-depth manner, enabling a thorough evaluation of the system by examining specific details and components and delving into finer aspects and intricacies of the architecture.

Figure 4.8 represents the detailed form of the architecture.

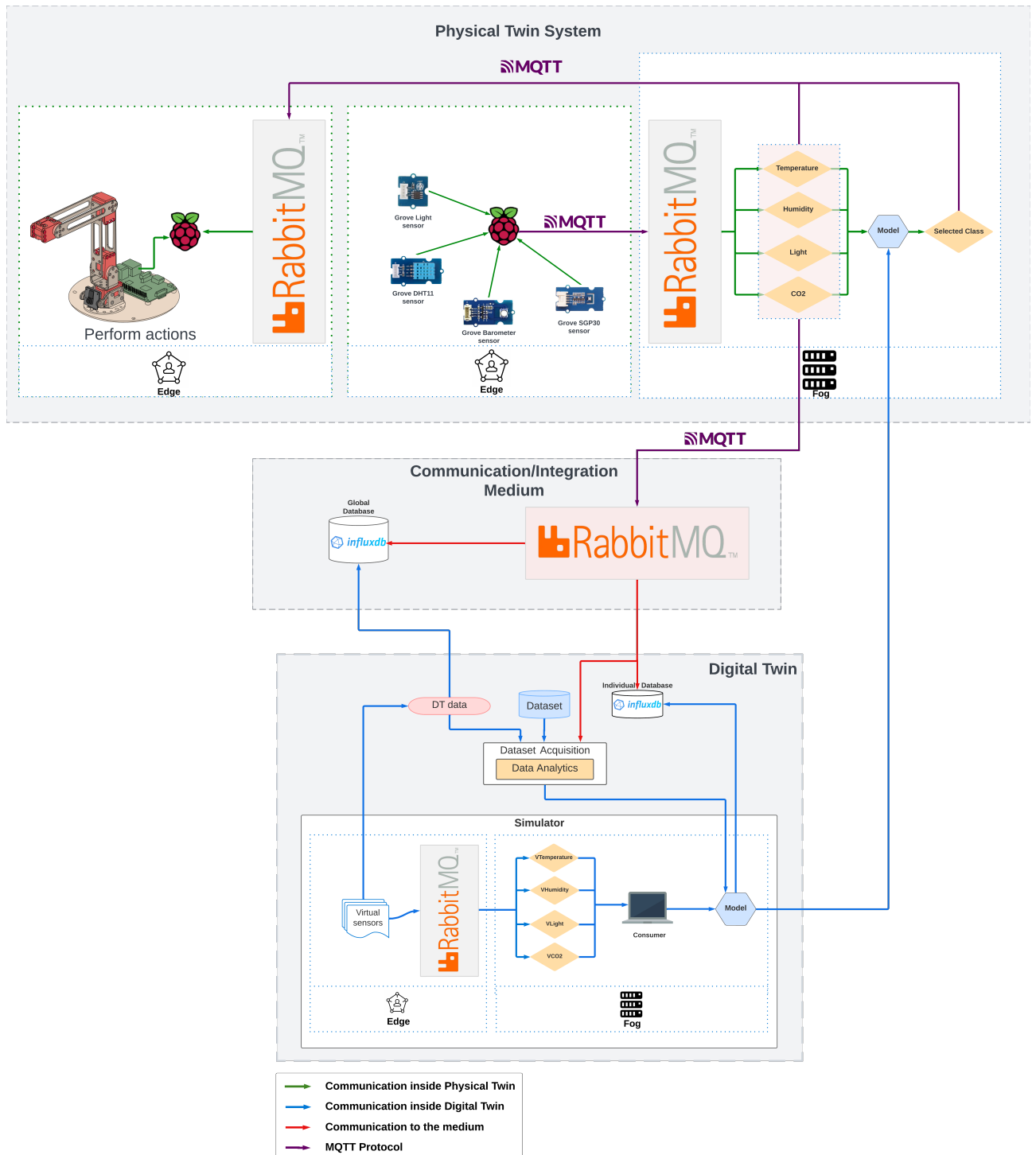


Figure 4.8: The system's architecture in a detailed form.

## 4.2.1 Description of the Detailed Architecture

### 4.2.1.1 Physical Twin

- **The First sub-system:** This first sub-system is in the form of an Edge/Fog architecture:

- PTEdge1 - Sensors: this edge consists of a Raspberry Pi 3 Model B connected to it four types of sensors which are:
  - \* Grove Temperature Sensor (Barometer).
  - \* Grove Humidity Sensor (DHT11).
  - \* Grove Light Sensor.
  - \* Grove CO2 Sensor (SGP30).

The data collected from the environment using these sensors are sent to the RabbitMQ broker located in the PTFog using MQTT protocol.

- PTFog: The PTFog receives the data sent by the PTEdge periodically and use them to be the input of the previously trained model on the Occupancy dataset. When the class is predicted (either Occupied or not Occupied), the data and the prediction is sent to the second sub-system and the Communication/Integration Medium using RabbitMQ and the same protocol (MQTT).
- **The Second sub-system:** this second edge is the OS of the poppy and the poppy itself. When the data is received, The poppy performs different actions depending on it, so that afterwards a decision-making process can be made.

### 4.2.1.2 Communication Medium

The communication or integration medium of a DT is typically a broker that makes the communication between the PT and DT safer. It provides a bi-directional communication and saves all the interactions between the PT and its different DT. In our use case, only one DT is created.

#### 4.2.1.3 Digital Twin

- **Dataset Acquisition Module:** As explained, a DT receives data from different sources, in this architecture, the different sources are shown which are:
  - The external dataset (Occupancy Dataset).
  - The data sent from the PT.
  - The data generated from the DT Simulator.
- The PT data is saved in an Individual InfluxDB dataset.
- **Simulator:** the core of the DT is in the form of an edge/fog to replicate perfectly the functioning of the first PT sub-system.
  - DTEdge 1 - VSensors: The DT Edge has virtual sensors that generates data similar to the PT's as well as disturbances. Using RabbitMQ MQTT, the generated data is sent to its corresponding Fog.
  - DTFog: Similar to the PT Fog, it receives the data from its edge and use it as an input to the model. What differs between the two models is that the DT model is aware of the disturbances and the PT only classifies the data as occupied and not occupied. This DTModel do a multi-class classification and when the prediction is made, it is saved in the individual InfluxDB database and global InfluxDB database.



### **4.3 Elaborated Architecture of the System**

This subsection presents an expanded view of the architecture, offering a detailed and extensive analysis of its components and functionality. It involves a comprehensive exploration of the architecture's intricate details, providing a comprehensive understanding of its various aspects. It encompasses a meticulous examination of each element, allowing for a comprehensive and nuanced perspective on the subject matter or situation.

Figure 4.9 represents the elaborated form of the architecture.

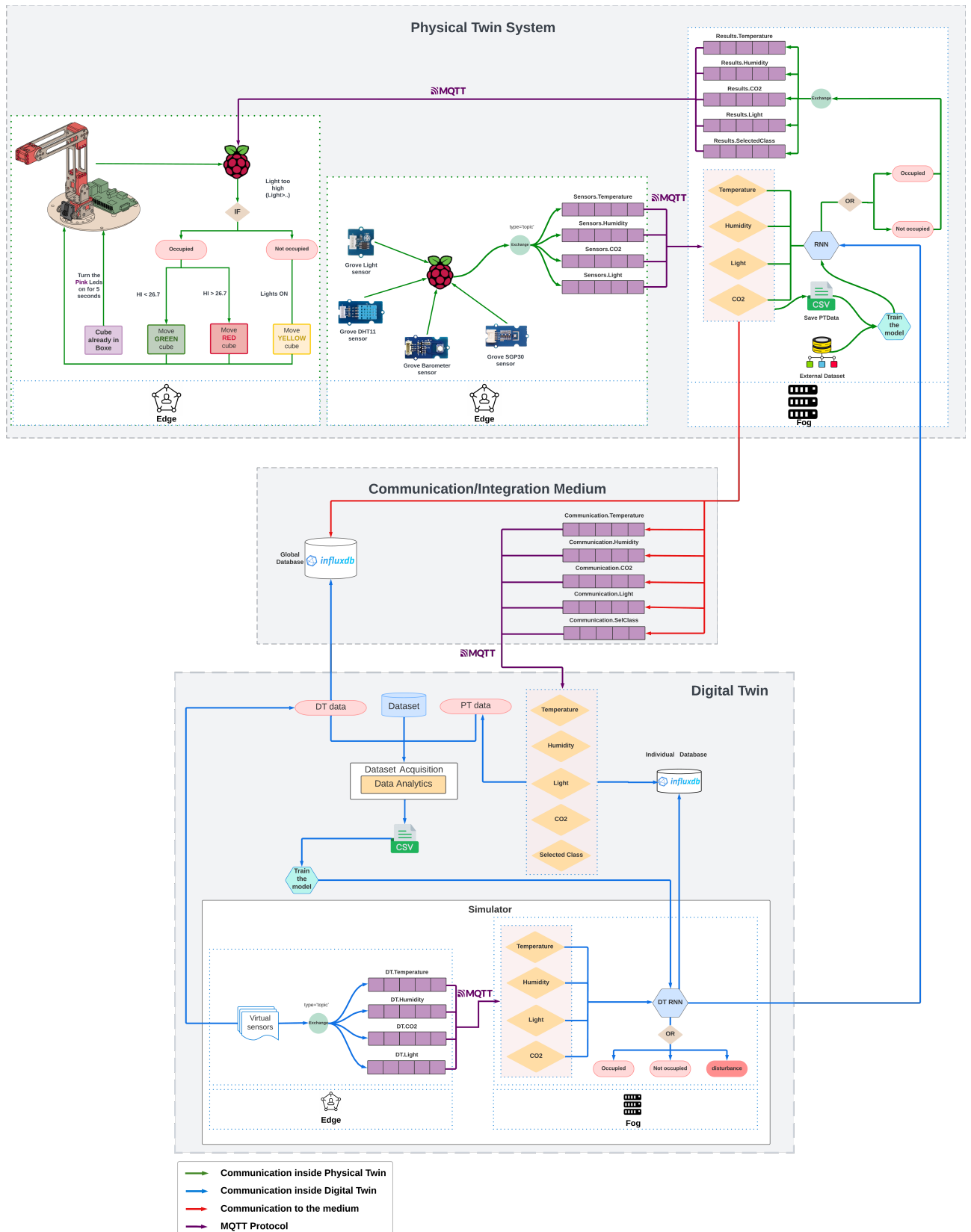


Figure 4.9: The system's architecture in an elaborated form.

### 4.3.1 Description of the Elaborated Architecture

#### 4.3.1.1 Physical Twin

- **The First sub-system:** This first sub-system is in the form of an Edge/Fog architecture:
  - PTEdge1 - Sensors: this edge consists of a Raspberry Pi 3 Model B connected to it a GrovePi+ add-on which is an adapter to the four types of sensors. Figure 4.10 represents the interconnection of everything together. The data collected from the environment using these sensors are sent to the RabbitMQ broker located in the PT-Fog using MQTT protocol with four different topics that have the following routing keys "Sensors.Temperature", "Sensors.Humidity", "Sensors.Light" and "Sensors.CO2".
  - PTFog: The PTFog receives the data sent by the PTEdge periodically and use them to be the input of the previously trained model on the Occupancy dataset at first. Meanwhile, the received data is saved in a CSV file that is used later on to train a new, more preferment model. When the class is predicted (either Occupied or not Occupied), the data and the prediction is sent to the second sub-system and the Communication/Integration Medium using RabbitMQ and the same protocol (MQTT). The routing keys of the topics from the PTFog to the second edge are: "Results.Temperature", "Results.Humidity", "Results.Light", "Results.CO2" and "Results.SelectedClass".

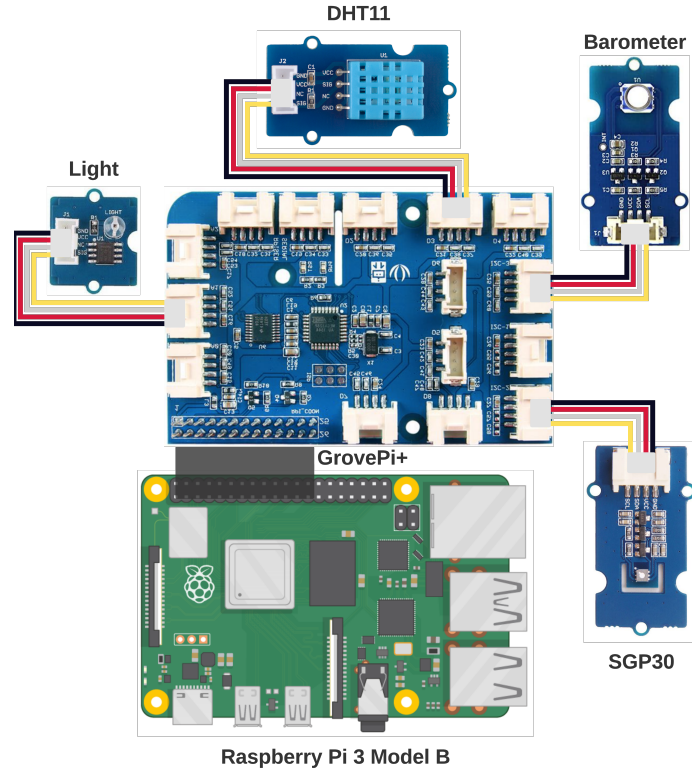


Figure 4.10: The Physical Twin's Edge Architecture.

- **The Second sub-system:** this second edge is the OS of the poppy and the poppy itself. When the data is received, The poppy performs different actions depending on it, so that afterwards a decision-making process can be made. The actions are the following:
  - When the room is occupied, the Heat Index (HI) is calculated, which is a measure that combines temperature and relative humidity to determine how hot it feels to the human body. The formula of the HI is presented in equation 4.1.

$$\begin{aligned}
HI &= c_1 + c_2 \cdot T + c_3 \cdot RH + c_4 \cdot T \cdot RH \\
&+ c_5 \cdot T^2 + c_6 \cdot RH^2 + c_7 \cdot T^2 \cdot RH \\
&+ c_8 \cdot T \cdot RH^2 + c_9 \cdot T^2 \cdot RH^2
\end{aligned} \tag{4.1}$$

$$where : \tag{4.2}$$

$HI$  is the Heat Index

$T$  is the temperature in Celsius

$RH$  is the relative humidity in percentage

$$c_1, c_2, \dots, c_9 \text{ are the coefficients specific to the equation} \tag{4.3}$$

The Variables in 4.4 represent the commonly used coefficients.

$$\begin{aligned}
c_1 &= -8.78469475556 \\
c_2 &= 1.61139411 \\
c_3 &= 2.33854883889 \\
c_4 &= -0.14611605 \\
c_5 &= -0.012308094 \\
c_6 &= -0.0164248277778 \\
c_7 &= 0.002211732 \\
c_8 &= 0.00072546 \\
c_9 &= -0.000003582
\end{aligned} \tag{4.4}$$

\* If the HI calculated is too high, then the red cube is displaced.

The interpretation of this is "*Open windows, Turn on the air-conditioners and use cooling measures*".

\* If the HI is low, move green cube, its corresponding interpretation

is *Adapt the air-conditioning*

- If the room is not occupied and the lights are ON then the yellow cube is moved. The interpretation is "*Turn the lights OFF*".

Figure 4.11 visually depicts the physical appearance of the second sub-system.

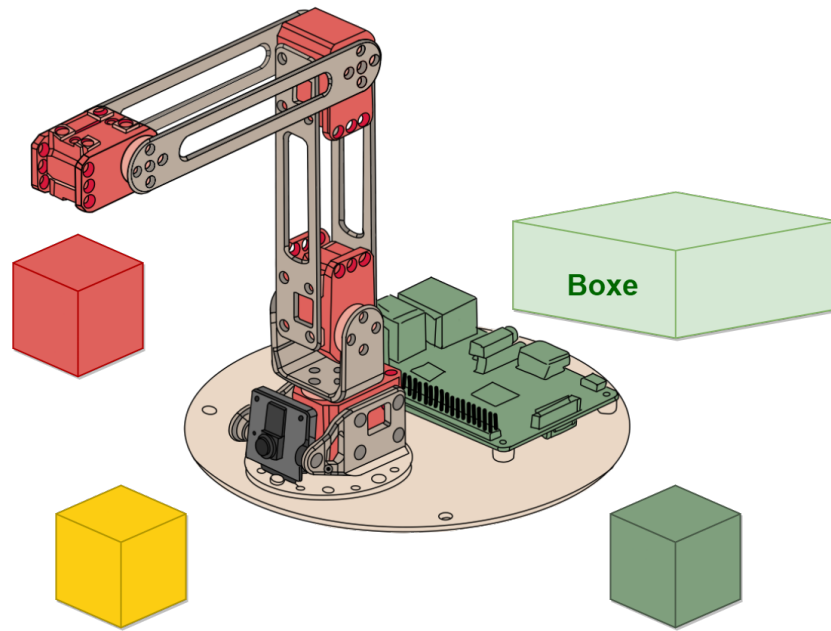


Figure 4.11: The Physical Twin's Edge Architecture.

#### 4.3.1.2 Communication Medium

The communication or integration medium of a DT is typically a broker that makes the communication between the PT and DT safer. It provides a bi-directional communication and saves all the interactions between the the PT and its different DT. In our use case, only one DT is created. the routing keys of the topics are:

- Communication.Temperature

- Communication.Humidity
- Communication.Light
- Communication.CO2
- Communication.SelClass

#### 4.3.1.3 Digital Twin

- **Dataset Acquisition Module:** As explained, a DT receives data from different sources, in this architecture, the different sources are shown which are:
  - The external dataset (Occupancy Dataset).
  - The data sent from the PT.
  - The data generated from the DT Simulator.
- The PT data is saved in an Individual InfluxDB dataset.
- The resulted global dataset is saved in a CSV file and that would be used to train the DT model.
- **Simulator:** the core of the DT is in the form of an edge/fog to replicate perfectly the functioning of the first PT sub-system.
  - DTEdge 1 - VSensors: The DT Edge has virtual sensors that generates data similar to the PT's as well as disturbances. Using RabbitMQ MQTT, the generated data is sent to its corresponding Fog with topics that has theese routing keys "DT.Temperature", "DT.Humidity", "DT.CO2" and "DT.Light".
  - DTFog: Similar to the PT Fog, it receives the data from its edge and use it as an input to the model.to make predictions. At a certain moment, the DT model replaces thePT model so that the PT would be aware of the disturbances.

## 5 Implementation

### 5.1 Class Diagram

The realised system is based on the class diagram represented in Figure 4.12. It represents all the implemented classes used to run the system as well as some classes that will be implemented.

#### 5.1.1 Description of the Class Diagram

The model is composed of the following classes:

- **Class "Component"**

The approach of the class diagram is to follow a System of Systems (SoS) Structure. A SoS is a collection of multiple, independent systems that are part of a larger, more complex system [73]. The constituent systems pool their resources and capabilities together to create a new, more complex system that offers more functionality and performance than simply the sum of the constituent systems [74]. SoSes enable the creation and operation of large and complex systems like the IoT system we are dealing with. The class "Component" presented in Figure 4.13 is at the head of the class diagram because everything is a component going from a complex to a simpler class.

It has the attributes "credentials", "broker", "port" that would be shared over the other classes and an operation "runComponent()" that would be overridden by other classes.

Two classes inherit from it:

- Class "PhysicalTwin".
- Class "DigitalTwin".



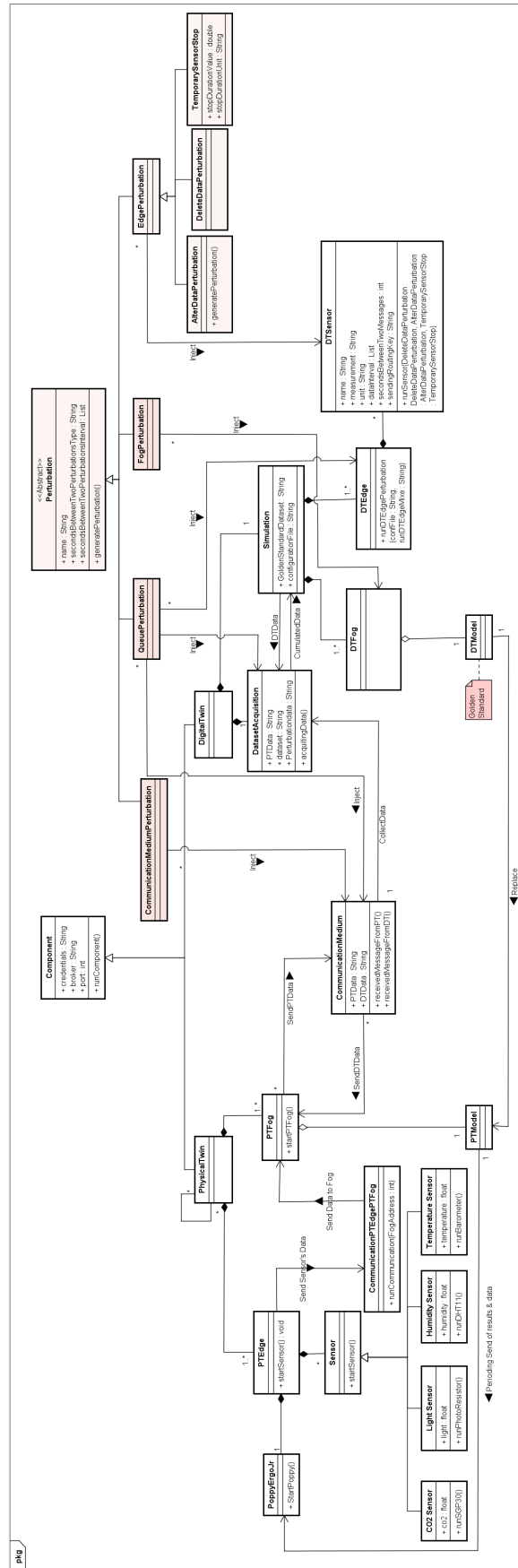


Figure 4.12: The System's Architecture in the Form of a Class Diagram

“The Use of Cognitive Digital Twins on an IoT System for Edge Resilience and 109 Anomaly Detection” Engineering Thesis

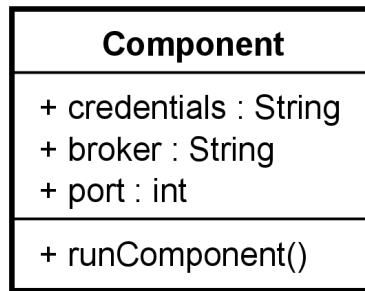


Figure 4.13: Class "Component".

- **Class "Physical Twin"**

This class, represented in Figure 4.14, is inherited from the class presented previously, class "Component" and overrides the **runComponent()** function. It is composed of two class:

- Class "PTEdge".
- Class "PTFog".

It can be composed of one to several PTEdge and one to several PTFog. The class "Physical Twin" is composed of itself referring to the SoS principle.

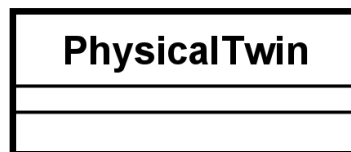


Figure 4.14: Class "Physical Twin".

- **Class "PTEdge"**

This class (Figure 4.15) is one of the compositions of the "physical twin" class, it represents the hardware of the system and has an operation

"startSensor()" to start the sensors, that is why it has a relation of composition with these two classes:

- Class "Sensor".
- Class "PoppyErgoJr".

It can be composed of several sensors. But it is composed of only one Robot Poppy Ergo Jr.

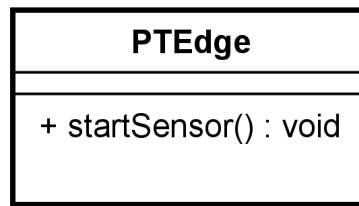


Figure 4.15: Class "Physical Twin's Edge".

- **Class "Sensor"**

This class represents all the sensors of the used use case. Figure 4.16 showcases it and four classes inherit from it:

- Temperature Sensor.
- Humidity Sensor.
- Light Sensor.
- CO2 Sensor.

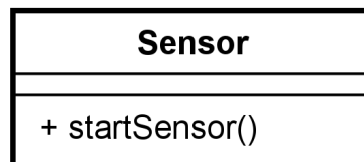


Figure 4.16: Class "Physical Twin's Sensor".

- **Class "Temperature Sensor"**

The class "Temperature" (Figure 4.17) represents the Barometer sensor, even though the other used sensor which is DHT11 sensor collects the humidity as well as the temperature, another sensor is used to prove that the created simulator is extensible.

the Temperature class has an operation "runBarometer()" that collects the temperature of the environment and put the value in its float attribute temperature.

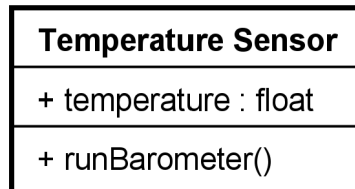


Figure 4.17: Class "Temperature Sensor".

- **Class "Humidity Sensor"**

Similar to the Temperature class, Figure 4.18 shows that the Humidity class has an operation "runDHT11()" that collects the humidity of the environment and put the value in its float attribute humidity.

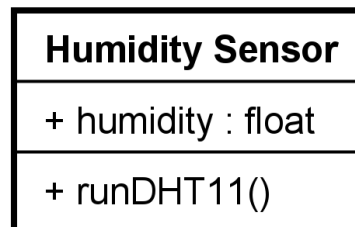


Figure 4.18: Class "Humidity Sensor".

- **Class "Light Sensor"**

Resembling the two previous classes, this class Light (Figure 4.19) has an operation "runPhotoResistor()" that captures the light in Lux and saves the value in its corresponding attribute "light".

Light Sensor
+ light : float
+ runPhotoResistor()

Figure 4.19: Class "Light Sensor".

- **Class "CO2 Sensor"**

This class, presented in Figure 4.20 has an attribute co2 that gets its value from the runGPS30() operation. runGPS30 runs the sensor and collect the corresponding value and saves it.

CO2 Sensor
+ co2 : float
+ runSGP30()

Figure 4.20: Class "CO2 Sensor".

- **Class "PoppyErgoJr"**

The "PoppyErgoJr" class represents the Robot arm named Poppy Ergo Jr (Figure 4.21). It has an operation "startPoppy()" that starts the robot and make actions depending on the data it receives periodically from the fog.

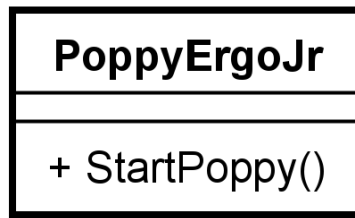


Figure 4.21: Class "Poppy Ergo Jr".

- **Class "CommunicationPTEdgePTFog"**

This class serves as a communication medium. It is shown in Figure 4.22 which shows that it has a function "runCommunication()". This operation sends the data collected by the edge (PTedge and its sensors) to the fog (class "PTFog") for processing.

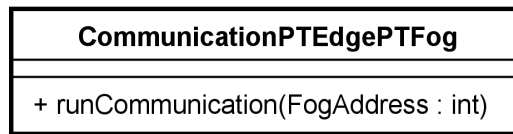


Figure 4.22: Class "Poppy Ergo Jr".

- **Class "PTFog"**

Figure 4.23 represents the PTFog class. It has a function "startPTFog()" that permits to receive the data from the "CommunicationPTEdgePTFog" class. After receiving those data, a model is trained and those same data are sent to the "CommunicationMedium" class.

The "PTFog" receives the generated data of the Digital twin as well.

- **Class "PTModel"**

This class (Figure 4.24) represents the model trained in the PTFog, and that would be used by the "PoppyErgoJr" class.

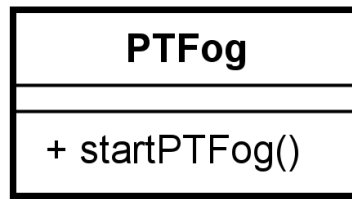


Figure 4.23: Class "PTFog".

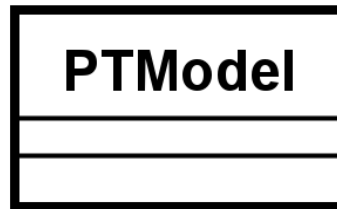


Figure 4.24: Class "PTModel".

- **Class "CommunicationMedium"**

This class (Figure 4.25), permits the bidirectional communication between the Physical Twin and Digital Twin by sending and receiving their corresponding data.

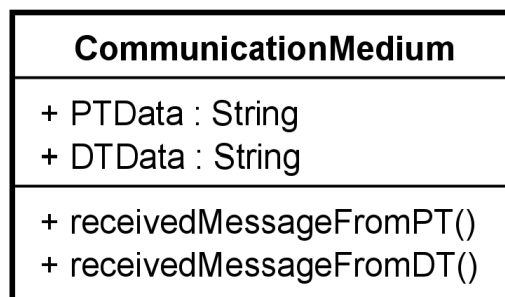


Figure 4.25: Class "CommunicationMedium".

- **Class "Digital Twin"**

This class, represented in Figure 4.26, is inherited from the class "Component" and overrides the `runComponent()` function. It is composed of

two class:

- Class "DataAcquisition".
- Class "Simulation".



Figure 4.26: Class "Digital Twin".

• **Class "DatasetAcquisition"**

The datasetAcquisition class represented in Figure 4.27 is a composition of the Digital Twin class and handle different sources of the Digital Twin which are its attributes:

- PTData.
- dataset.
- perturbationData.

It communicates with the class "Simulation".

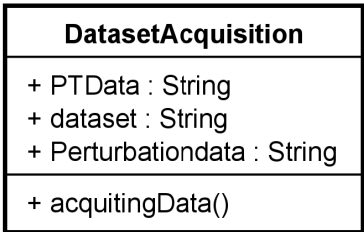


Figure 4.27: Class "DatasetAcquisition".

• **Class "Simulation"**



The class "Simulation" (Figure 4.28) is the head of the module that replicates the Physical Twin's functioning.

It receives the accumulated data from several sources received from the DatasetAcquisition module and reads the configuration file that has definitions of sensors which are both its attributes.

It is composed of these two classes:

- DTEdge.
- DTFog.

It can have on to several DTEdge and DTFogs.

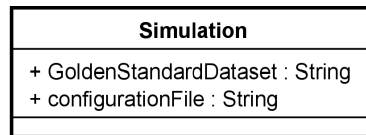


Figure 4.28: Class "Simulation".

- **Class "DTEdge"**

Figure 4.29 portrays the class "DTEdge", it is the replica of the PTEdge and has two operations:

- Operation "runDTEdgePerturbation()": this function generates only disturbances and saves it in a CSV file.
- Operation "runDTEdgeMixe()": this function generates sensor's data some that are altered and others that are not.

This class has a relation of composition with "DTSensor" class. It can be composed of several sensors.

- **Class "DTSensor"**

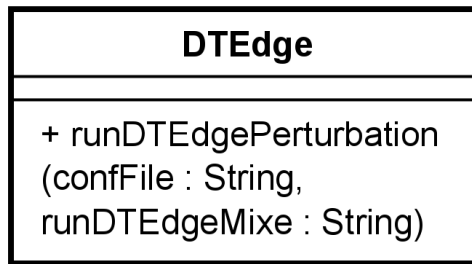


Figure 4.29: Class "DTEdge".

To represent the digital version of a Sensor, many attributes has been given to the class DTSensor, as shown in Figure 4.30, this class has the following attributes:

- name.
- measurement.
- unit.
- dataInterval.
- secondsBetweenTwoMessages.
- SendingRoutingKey so that the runSensor method knows to which queue the message will be sent.

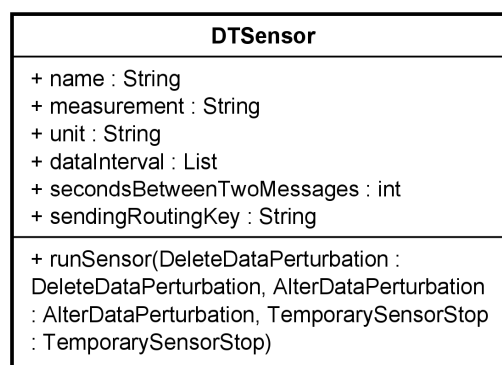


Figure 4.30: Class "DTSensor".

- **Class "DTFog"**

The class "DTFog" represented in Figure 4.31 has the same functionalities as the class "PTFog".

It trains a model using the Golden Standard Dataset.

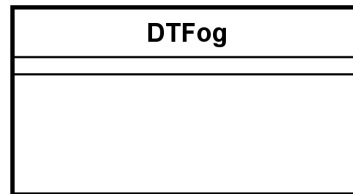


Figure 4.31: Class "DTFog".

- **Class "DTModel"**

Figure 4.32 represent the class "DTModel" that has the trained model, this model is considered to be the golden standard of this work.

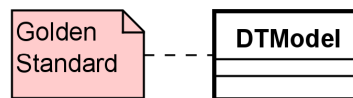


Figure 4.32: Class "DTModel".

- **Class "Perturbation"**

The abstract class "Perturbation" (Figure 4.33 has several children that are:

- Class "EdgePerturbation".
- Class "FogPerturbation".
- Class "CommunicationMediumPerturbation".
- Class "QueuePerturbation".

The type of disturbance that has been implemented is "EdgePerturbation".

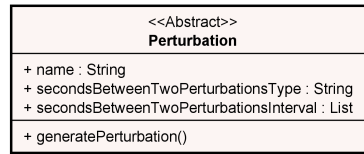


Figure 4.33: Class "Perturbation".

- **Class "Edge Perturbation"**

Figure 4.34 represents the class "Edge Perturbation" and its children which are:

- AlterDataPerturbation.
- DeleteDataPerturbation.
- TemporarySensorStop.

DeleteDataPerturbation class deletes the generated values, AlterDataPerturbation class changes the the generated values and TemporarySensorStop stops the sensor temporarily.

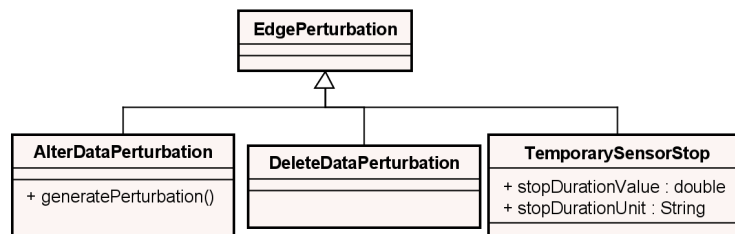


Figure 4.34: Classes of Edge Perturbation".

## 5.2 Creating a Dataset

As mentioned, the DT is trained on three different sources:

- the PT collected data.

- the external dataset.
- the altered and disturbed data generated by DT.

In this subsection, the external dataset and the created combined dataset are about to be analyzed.

### 5.2.1 The External Dataset

As presented previously, This external dataset represents data collected in a room (Temperature, Humidity, Light, CO2) in the range of approximately two weeks (from 03-02-2015 to 19-02-2015).

As shown in Figure 4.35 that represents a time series plot where the occupancy status can be visualized over time, the data has been collected when the room is occupied and when it is not to make a good distribution over time.

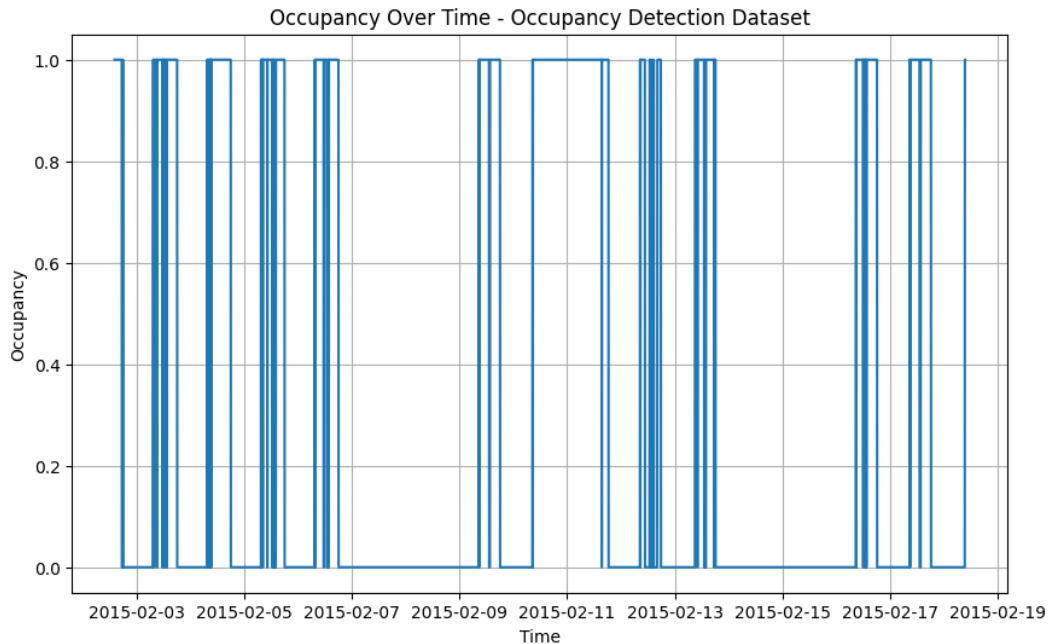


Figure 4.35: External Dataset - Time Series Plot

Figure 4.40 represents a correlation matrix that shows the relationships between different features which are the sensors.

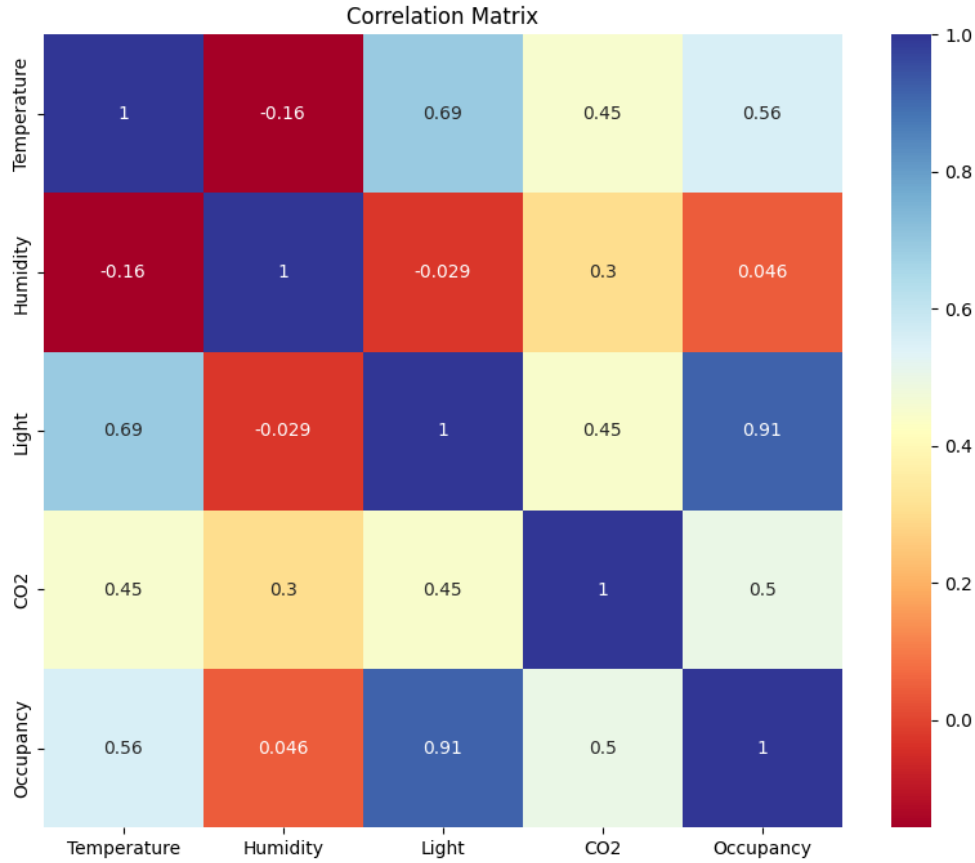


Figure 4.36: External Dataset - Correlation Matrix

By examining the correlation coefficients, what can be identified is that the Light variable is strongly correlated with occupancy.

Now, each feature is analysed individually.

- Histogram Temperature Plot:** in this diagram, the distribution of the Temperature sensor is displayed for occupied and unoccupied states. As it can be seen, there is a distinct pattern and few overlappings.

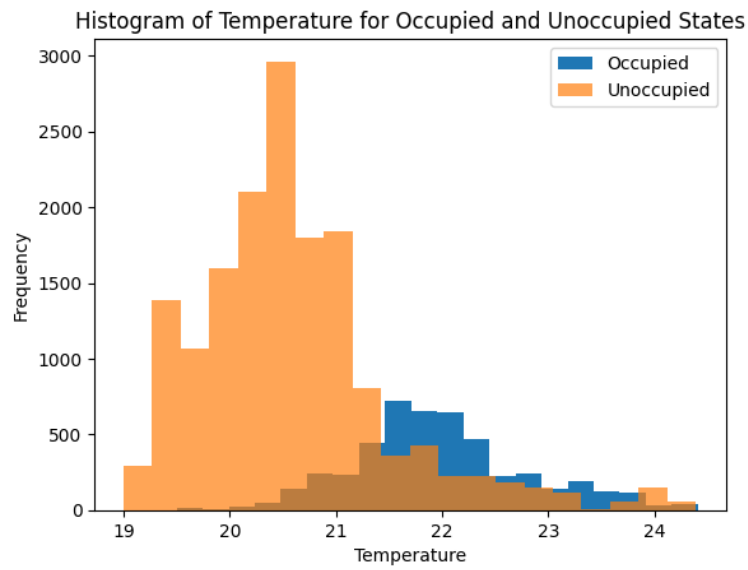


Figure 4.37: External Dataset - Temperature Histogram Plot

- **Histogram Humidity Plot:** There is more overlapping than distinction in the Humidity Histogram diagram.

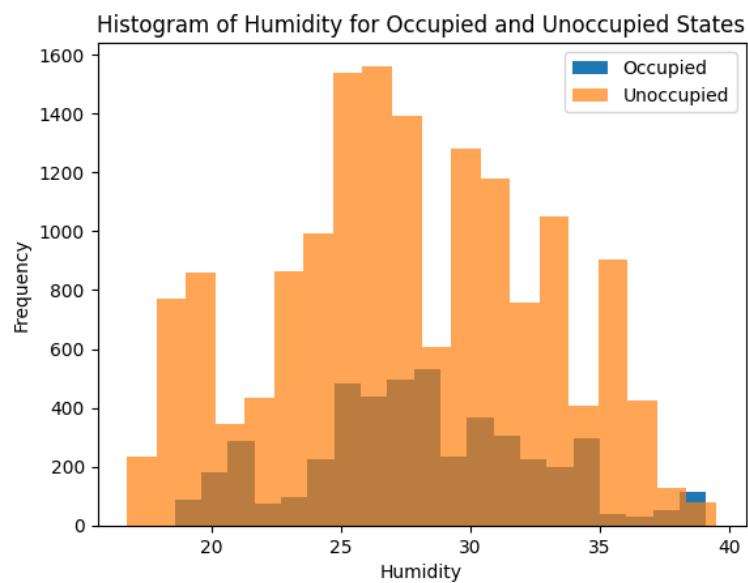


Figure 4.38: External Dataset - Humidity Histogram Plot

- **Histogram Light Plot:** The distinction is clear.

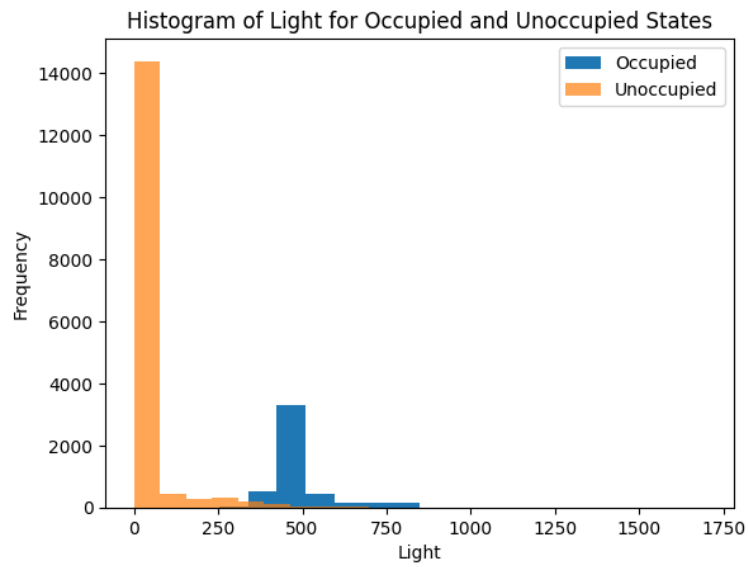


Figure 4.39: External Dataset - Light Histogram Plot

- **Histogram CO2 Plot:**

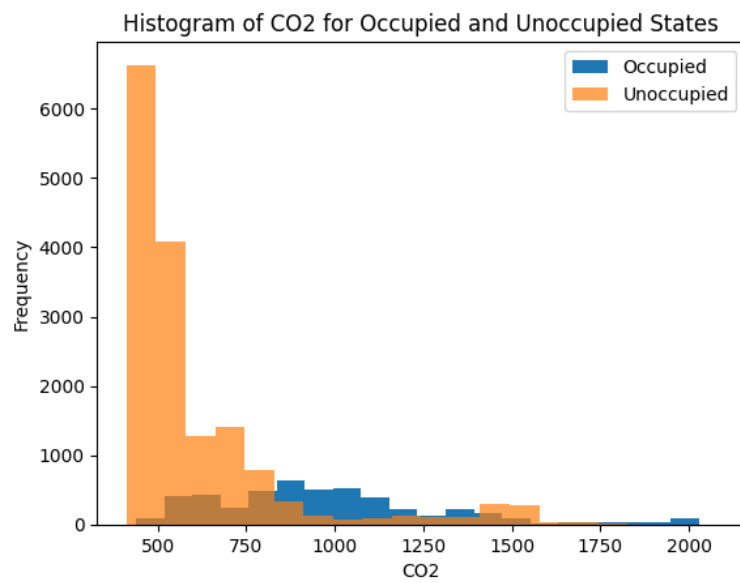


Figure 4.40: External Dataset - CO2 Histogram Plot

In general, the external dataset has a coherent distribution.



### 5.2.2 The Global Dataset

This global dataset represent the concatenation and combination of the three different sources (the external dataset "Occupancy Detection", PT's collected data, the disturbance generated by the DT).

The PT data has been collected in one day considered as the perfect day to collect the data. And as it can be seen in the graphs presented in Figure 4.41, which represent different histogram graphs of each feature/sensor, the global distribution is acceptable and it would not confuse the future trained model.

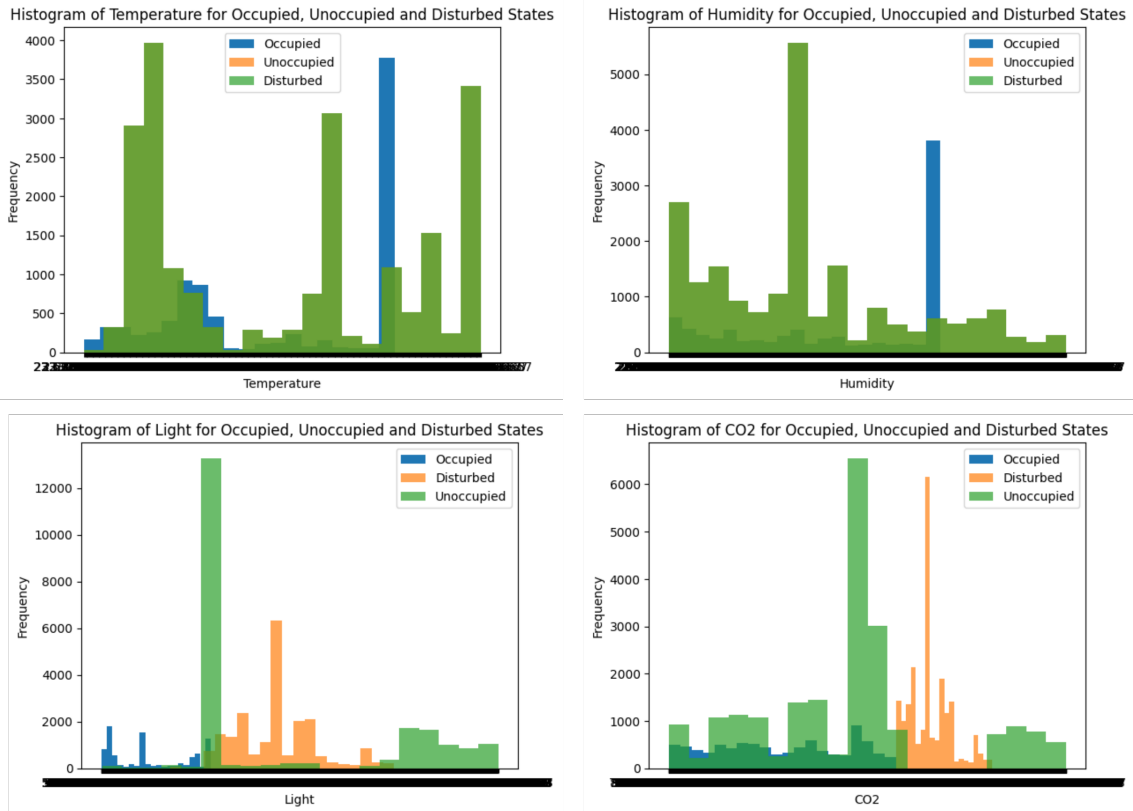


Figure 4.41: Global Dataset - Histogram Plots

## 5.3 Selecting an ML or DL Models

In this section, an adapted model would be trained for the PT and the DT, to find out which model among the ML and DL models is the most performing, different models have been trained which are:

- Decision Trees.
- Random Forests.
- KNN.
- Naive Bayes.
- RNN.
- MLP.

🔍 **Note:** The selection of the most performing hyper-parameters is done either with **GridSearch**<sup>3</sup> or **RandomSearch**<sup>4</sup>

### 5.3.1 The Physical Twin's Model

The PT's model goes through two steps:

- Select a model trained on the external dataset (Occupancy Detection Dataset), and test it on the data captured via the sensors which does not detect disturbances.
- Replace the model with a trained model in the DT so that it would give better results and acknowledge the disturbance.

Six different models have been trained on the external dataset and the results have been summarized in Figure 4.42.

---

<sup>3</sup>Grid search is a method for performing hyper-parameter optimization, which is used to find the optimal hyper-parameters of a model that results in the most accurate predictions

<sup>4</sup>Random search is a hyper-parameter optimization technique that involves selecting random combinations of hyper-parameters to train a model

As it can be seen, the F1 score of these models are quite close, that is why in the next subsections a few plots are presented.

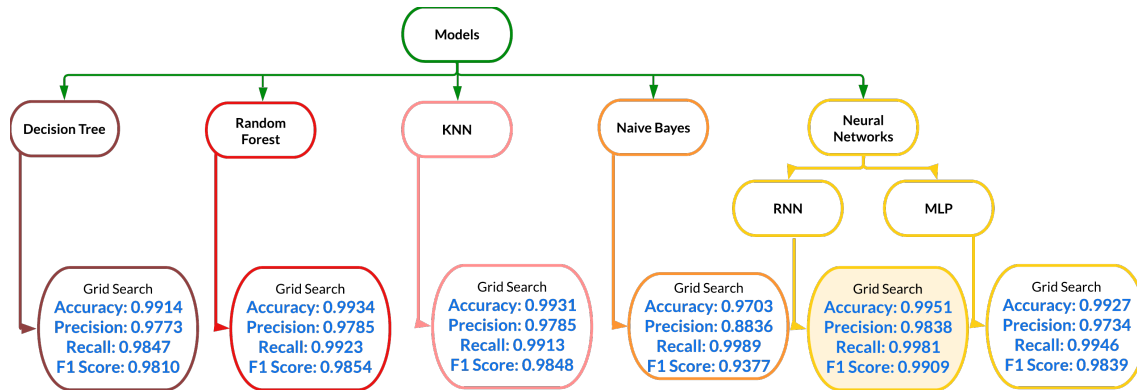


Figure 4.42: PT trained models on the external dataset

### 5.3.1.1 Decision Tree

The resulted decision tree is presented in Figure 4.43. and as shown in Figure 4.44, this model depends highly on the Light feature. It gave the following results:

- Accuracy: 99.14%
- Precision: 97.73%
- Recall: 98.47%
- F1 Score: 98.10%

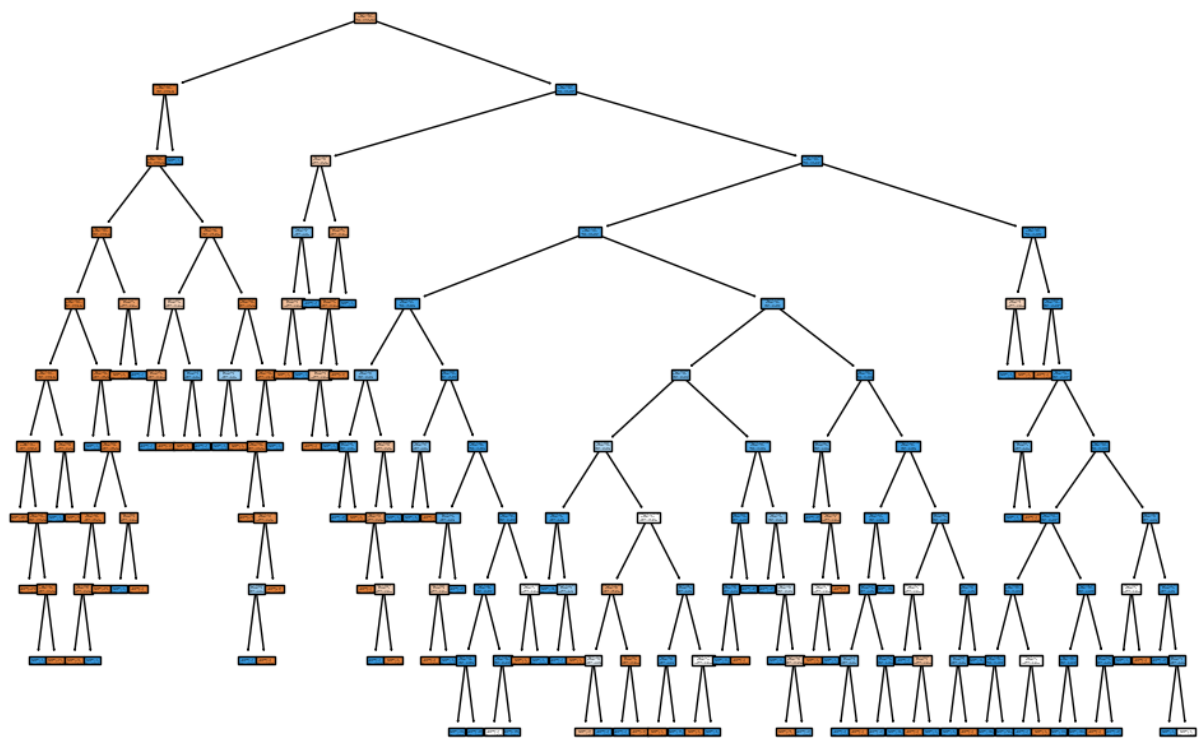


Figure 4.43: The resulted Decision Tree trained on the external dataset

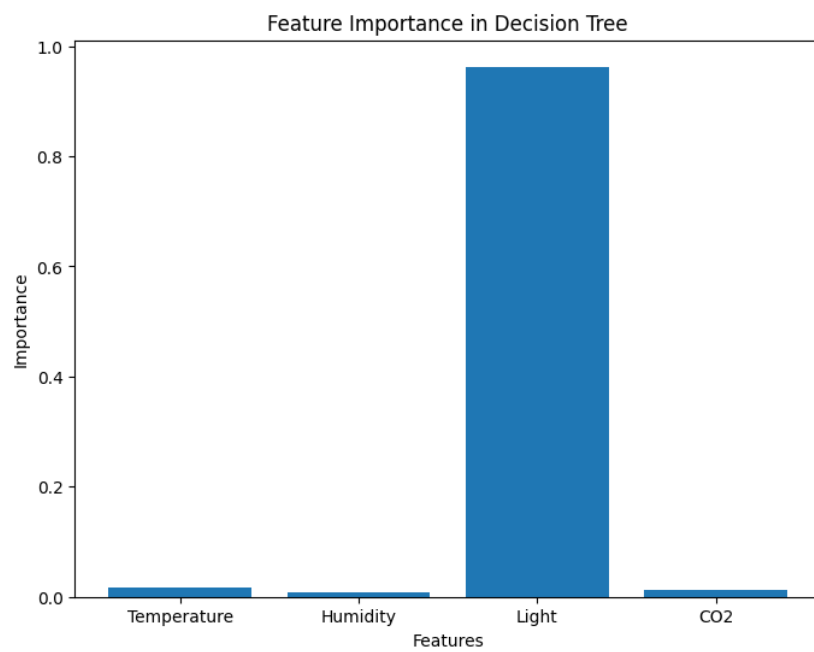


Figure 4.44: Feature Importance Plot - Decision Tree trained on the external dataset

### 5.3.1.2 Random Forests

Similar to the Decision Tree model and as it can be seen in Figure 4.45 that represents the feature importance plot, the light has a huge importance. However, it does give more importance to the Temperature and CO2 features.

It gave the following results, which are better than the Decision Tree Model:

- Accuracy: 99.34%
- Precision: 97.85%
- Recall: 99.23%
- F1 Score: 98.54%

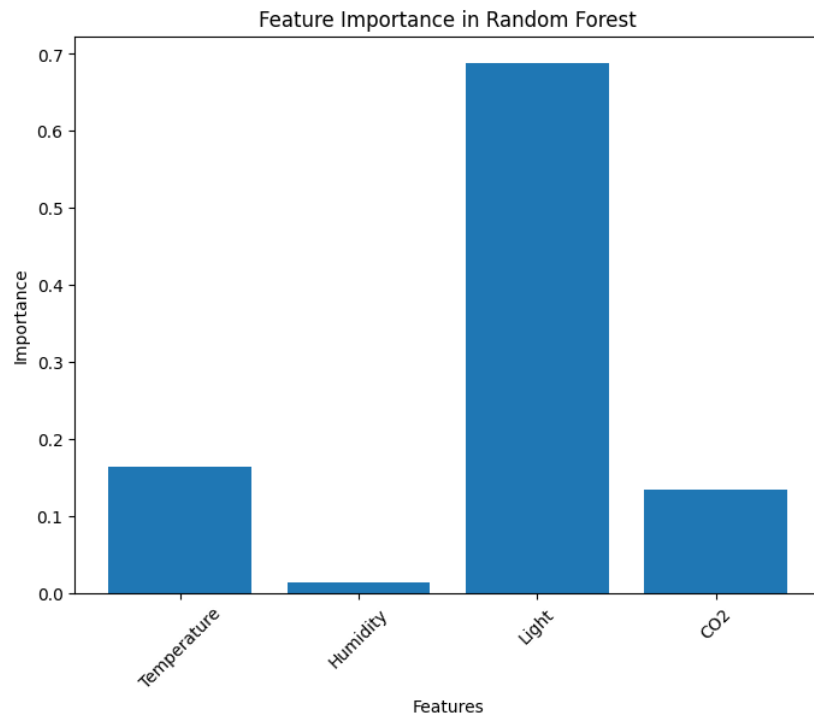


Figure 4.45: Feature Importance Plot - Random Forest trained on the external dataset

### 5.3.1.3 KNN

The RandomGridSearch selected the number of neighbors K to be "7" and the why is explained in Figure 4.46, which show that seven neighbors has the lowest error rate.

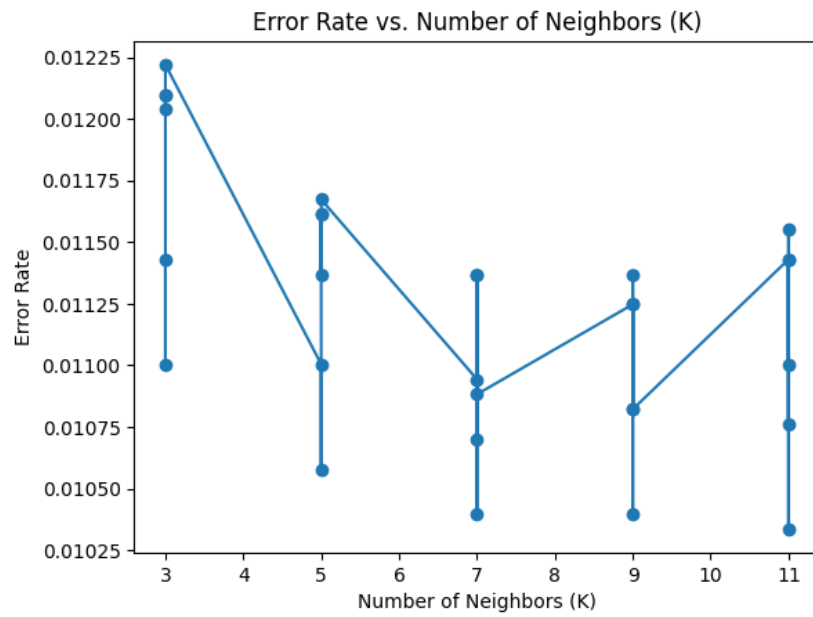


Figure 4.46: Error Rate vs Number of Neighbors (K)

In Figure 4.47, a corresponding Confusion Matrix is represented where the classifications can be seen:

- 3172 instances which are part of the class "Not Occupied" are correctly classified (True Negative TN).
- 20 instances which are part of the class "Occupied" are correctly classified (True Positive TP).
- 912 instances which are part of the class "Not Occupied" but are classified as "Occupied" (False Negative FN).

- 8 instances which are part of the class "Occupied" but are classified as "Not Occupied" (False Positive FP).

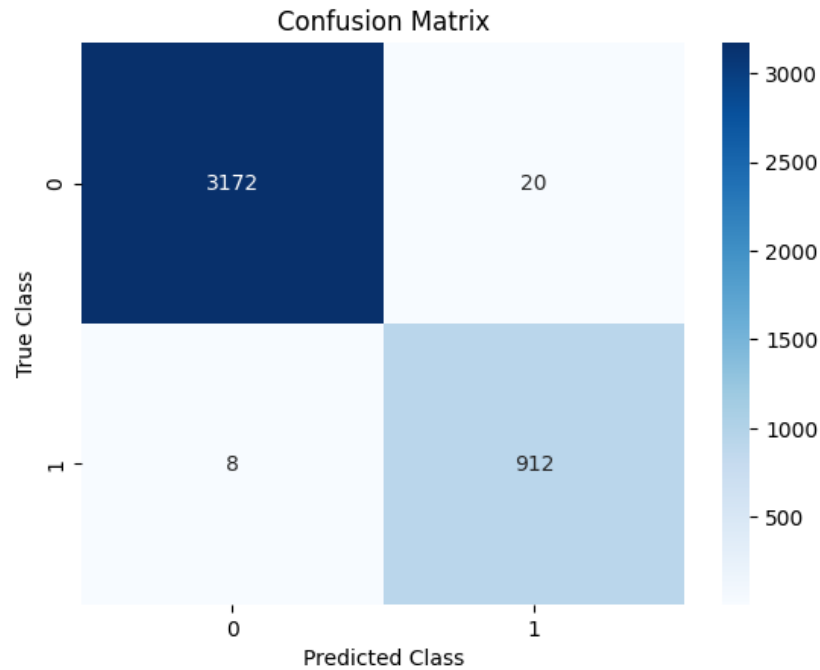


Figure 4.47: Confusion Matrix - KNN trained on the external dataset

It gave the following results, which are quite similar to the Random Forest Model:

- Accuracy: 99.31%
- Precision: 97.85%
- Recall: 99.13%
- F1 Score: 98.48%

#### 5.3.1.4 Naive Bayes

The first plot represented in Figure 4.48 shows Precision-Recall curves. It summarizes the trade-off between the TP rate and the positive predictive value.

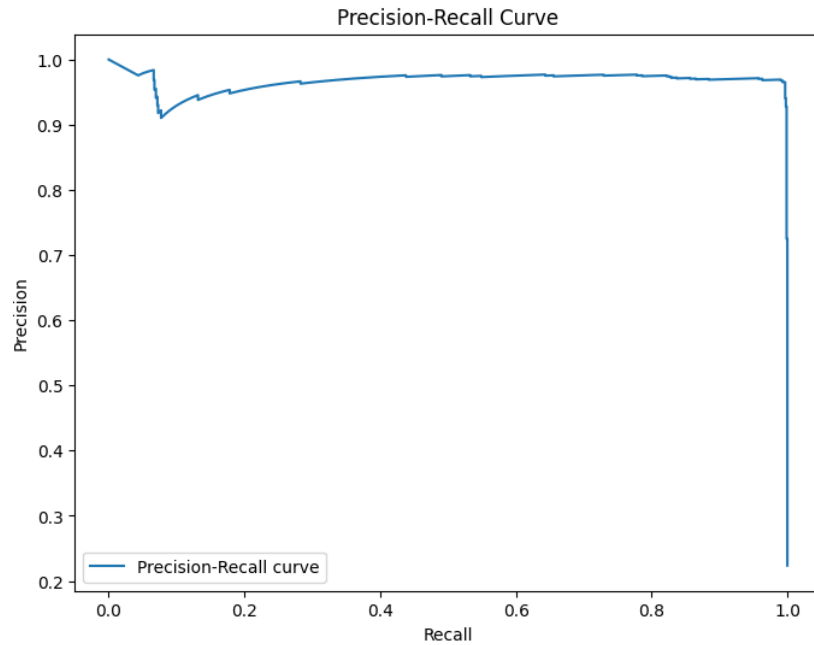


Figure 4.48: Precision-Recall Curve - Naive Bayes trained on the external dataset

Figure 4.49 is the confusion matrix of this model:

- 3071 instances which are part of the class "Not Occupied" are correctly classified (True Negative TN).
- 919 instances which are part of the class "Occupied" are correctly classified (True Positive TP).
- 121 instances which are part of the class "Not Occupied" but are classified as "Occupied" (False Negative FN).
- One instance which is part of the class "Occupied" but is classified as "Not Occupied" (False Positive FP).



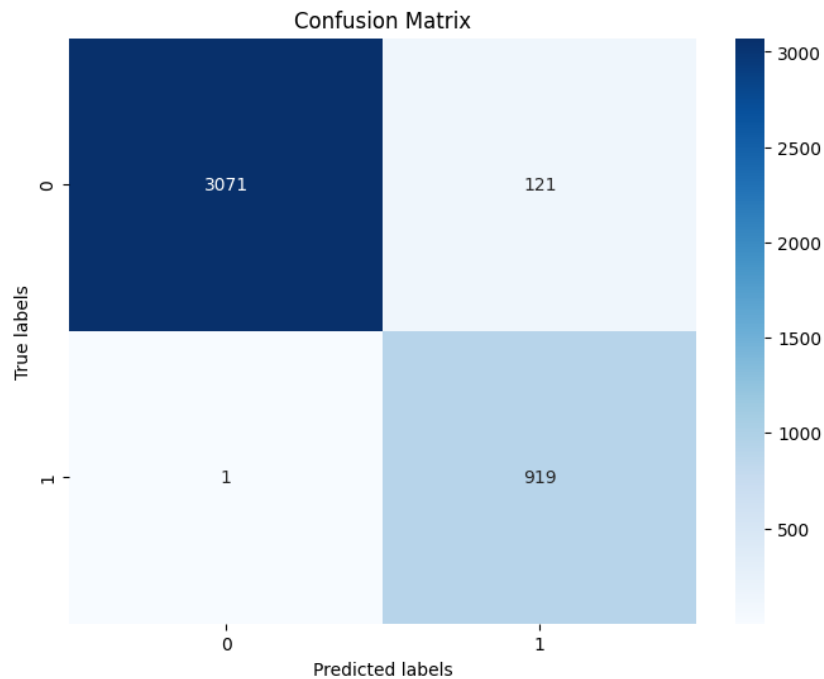


Figure 4.49: Confusion Matrix - Naive Bayes trained on the external dataset

It gave the following results, which are far less performing that the previously viewed models:

- Accuracy: 97.03%
- Precision: 98.36%
- Recall: 99.89%
- F1 Score: 93.77%

#### 5.3.1.5 MLP

The Figure 4.50, it can be seen the higher the iterations, the better the result is. and its confusion matrix in Figure 4.51 gives theses values:

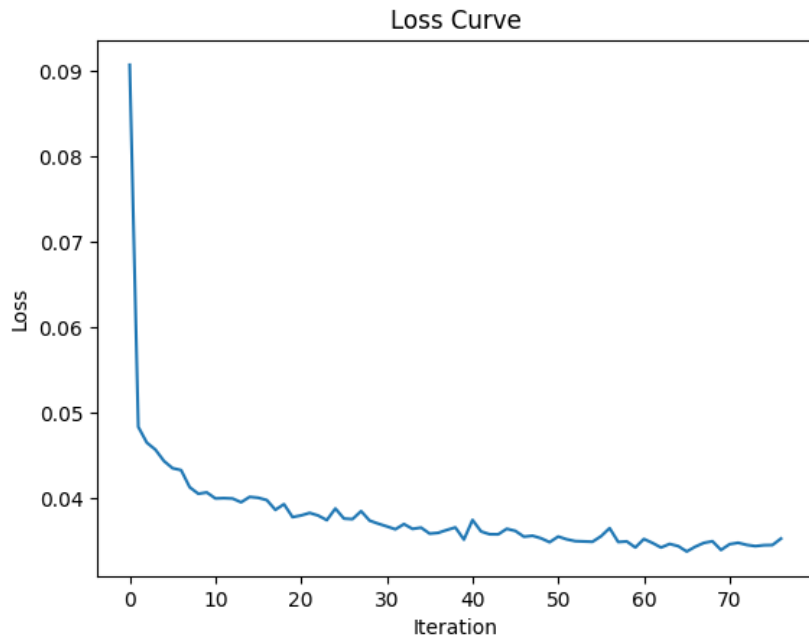


Figure 4.50: Loss Curve - MLP trained on the external dataset

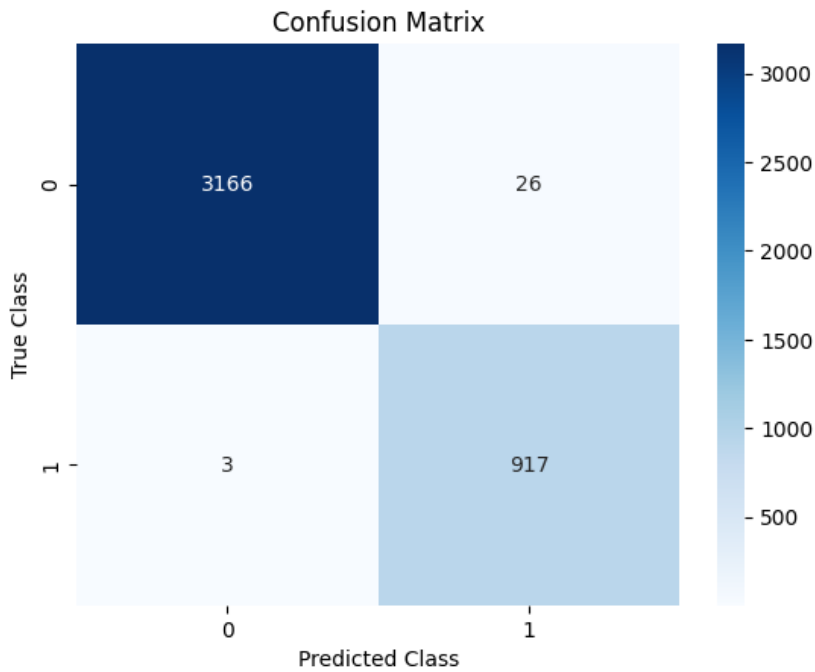


Figure 4.51: Confusion Matrix - MLP trained on the external dataset

- 3166 instances which are part of the class "Not Occupied" are correctly classified (True Negative TN).

- 917 instances which are part of the class "Occupied" are correctly classified (True Positive TP).
- 26 instances which are part of the class "Not Occupied" but are classified as "Occupied" (False Negative FN).
- 3 instances which are part of the class "Occupied" but are classified as "Not Occupied" (False Positive FP).

It gave the following results:

- Accuracy: 99.27%
- Precision: 97.34%
- Recall: 99.46%
- F1 Score: 98.39%

#### **5.3.1.6 RNN**

As seen in Figure 4.52, the best epochs iteration is 8 epochs. And the classification shown in the confusion matrix of Figure 4.53 are the following:

- 3132 instances which are part of the class "Not Occupied" are correctly classified (True Negative TN).
- 945 instances which are part of the class "Occupied" are correctly classified (True Positive TP).
- 30 instances which are part of the class "Not Occupied" but are classified as "Occupied" (False Negative FN).

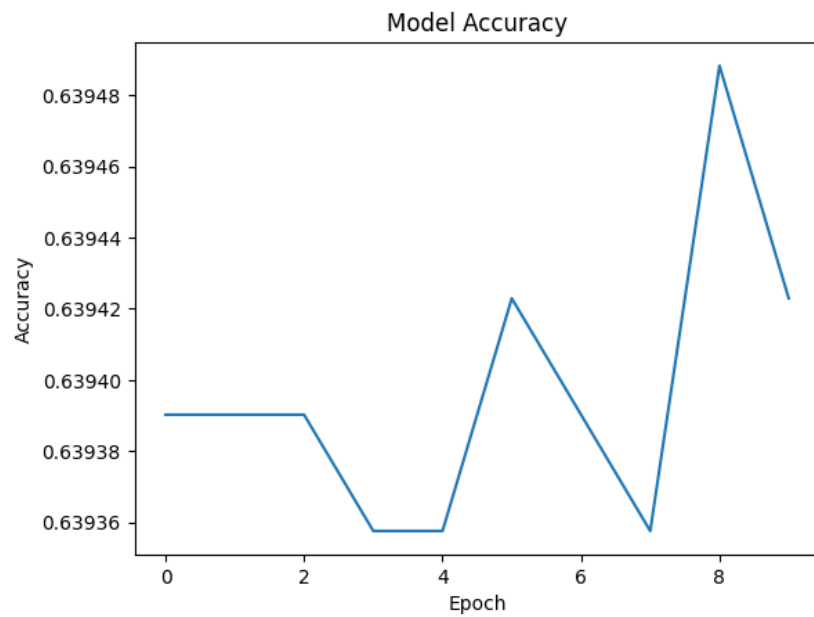


Figure 4.52: Feature Importance Plot - Decision Tree trained on the external dataset

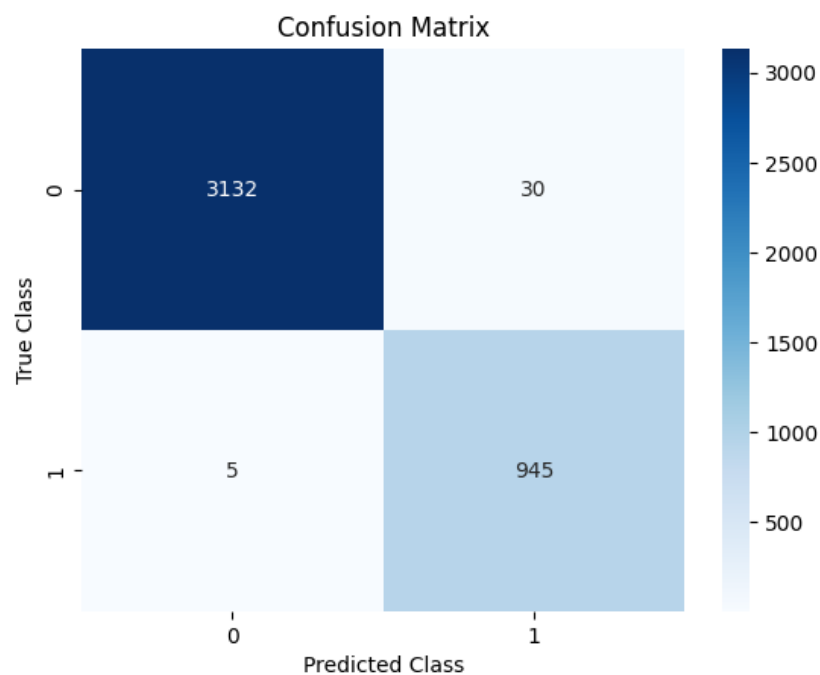


Figure 4.53: Confusion Matrix - RNN trained on the external dataset

- 5 instances which are part of the class "Occupied" but are classified as "Not Occupied" (False Positive FP).

It gave the following results:

- Accuracy: 99.51%
- Precision: 97.38%
- Recall: 99.81%
- F1 Score: 98.09%

#### 5.3.1.7 The PT Model selection

The selected model is the RNN due to its performance on the dataset and the fact that it is a type of DL models that is well-suited for time series data. As mentioned, it retains a memory of what it has already processed and can learn from previous iterations during its training.

Figure 4.54 shows the trained model which has the following characteristics:

- **LSTM Parameters:**
  - Unit: The number of LSTM units or cells in the layer is set to 64.
  - input\_shape: The shape of the input data for the LSTM layer is set to (1, n\_feature), where n\_feature is the number of features in this case five features (Timestamp, Temperature, Humidity, Light, CO2).
- **Model Training Parameters:**
  - optimizer: The optimizer used to update the weights during training is set to 'rmsprop'.
  - num\_dense\_layers: The number of dense layers after the LSTM layer are set to 4.
  - dropout\_rate: The dropout rate, which is the fraction of the input units to drop during training is set to 0.1.
  - dense\_units: The number of units/neurons in each dense layer is set to 16.

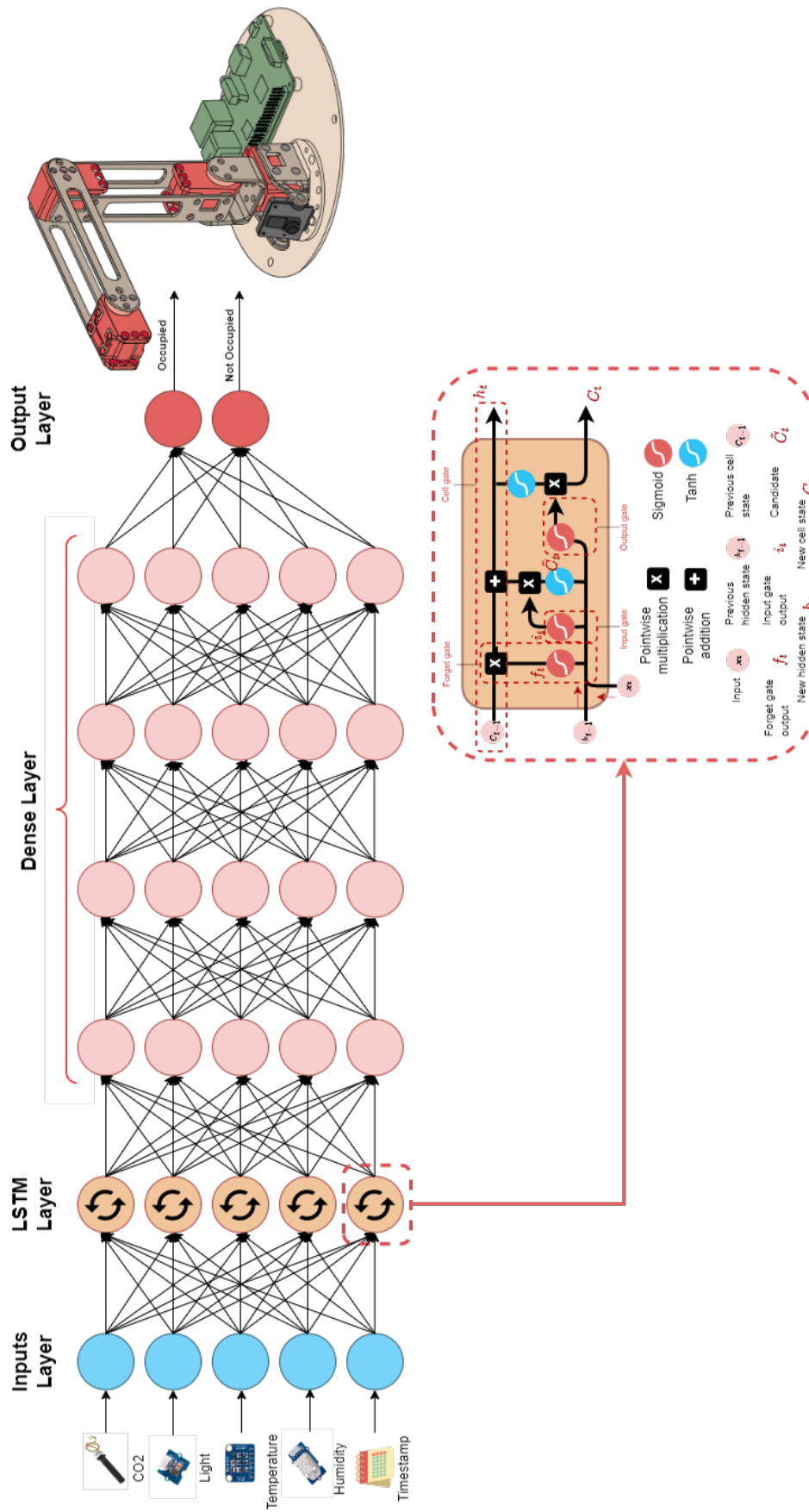


Figure 4.54: Physical Twin's Selected Model

"The Use of Cognitive Digital Twins on an IoT System for Edge Resilience and 138 Anomaly Detection" Engineering Thesis

- activation: The activation function used in the dense layers is set to 'sigmoid'.
- epochs: The number of times the model is trained on the entire training dataset is 10 iterations.
- batch\_size: The number of samples propagated through the network before the weights are updated is set to 32.

### 5.3.2 The Digital Twin's Model

The DT's model can be trained over the three sources of data and tested over the generated. As mentioned, the three sources are:

- The external dataset (Occupancy Dataset).
- The data sent from the PT.
- The data generated from the DT Simulator. The data is generated via a JavaScript Object Notation (JSON) file that describes the used sensors, this is the file which make the simulator expandable. The description of one of the sensors is presented in Figure 4.55.

Similar to the PT model, Six models have been trained on the global dataset:

- Decision Tree.
- Random Forest.
- KNN.
- Naive Bayes.
- Neural Networks.
- RNN.

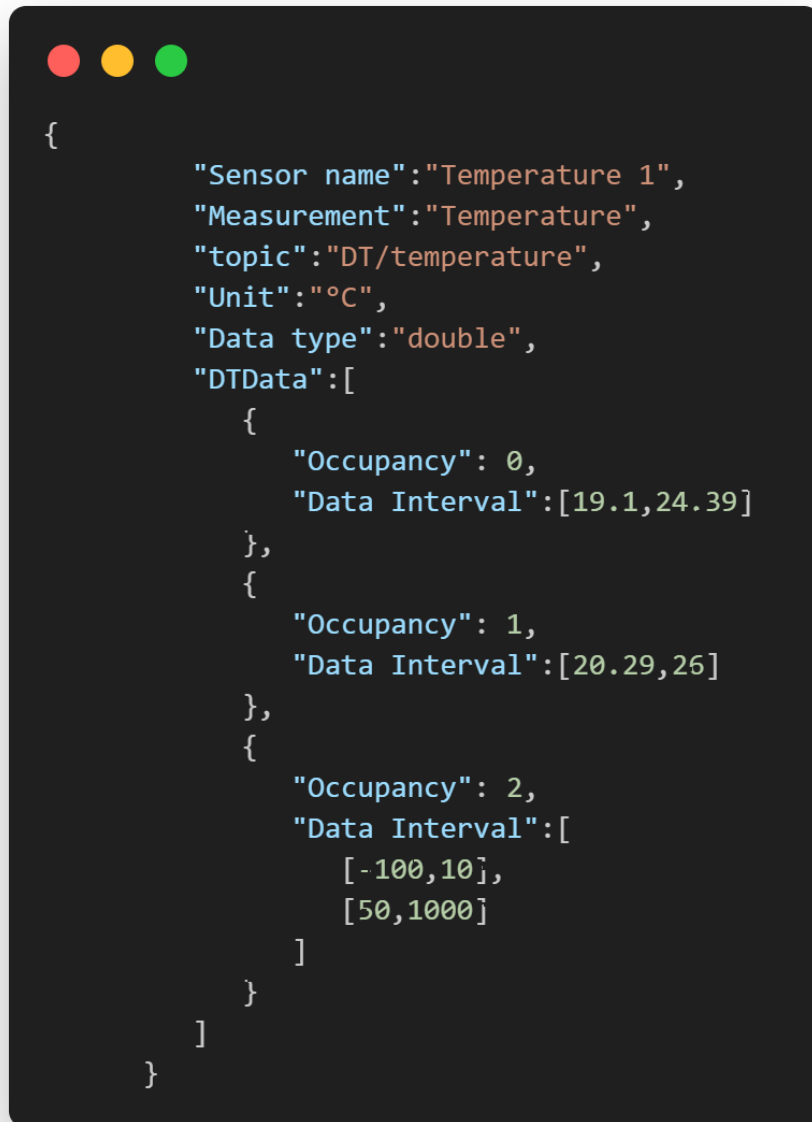


Figure 4.55: Description of the Temperature Sensor in the Digital Twin

– MLP.

The results have been resumed in Figure 4.56

Only well performing models will be discussed in the upcoming subsections.



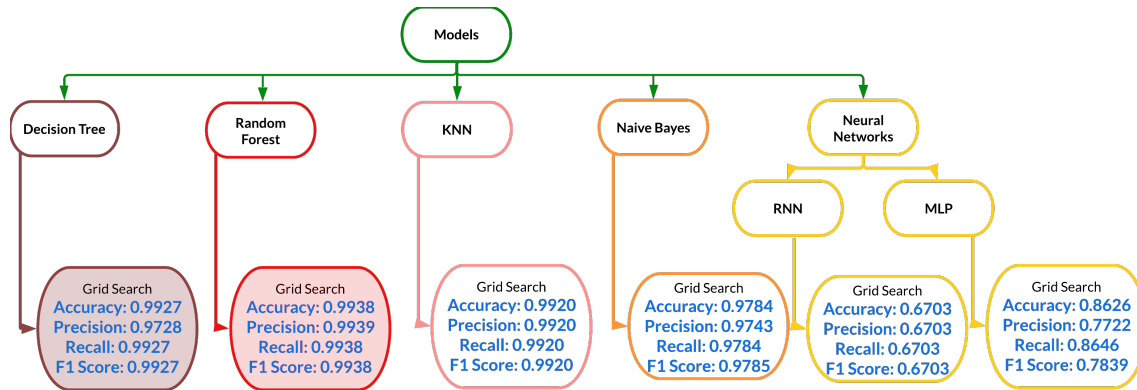


Figure 4.56: DT trained models on the global dataset

### 5.3.2.1 Decision Tree

The model gave very performing results and is presented in Figure 4.57.

And similar to the Occupancy Detection Dataset, the Light feature has a bigger importance than other features (Figure 4.58).

As shown in Figure 4.59, the binary problem became a multi-class problem and it has most of the classifications correct:

- 4194 instances which are part of the class "Not Occupied" are correctly classified.
- Only 7 instances which are part of the class "Not Occupied" are wrongly classified as "Occupied".
- No instances which are part of the class "Not Occupied" are wrongly classified as "disturbance".

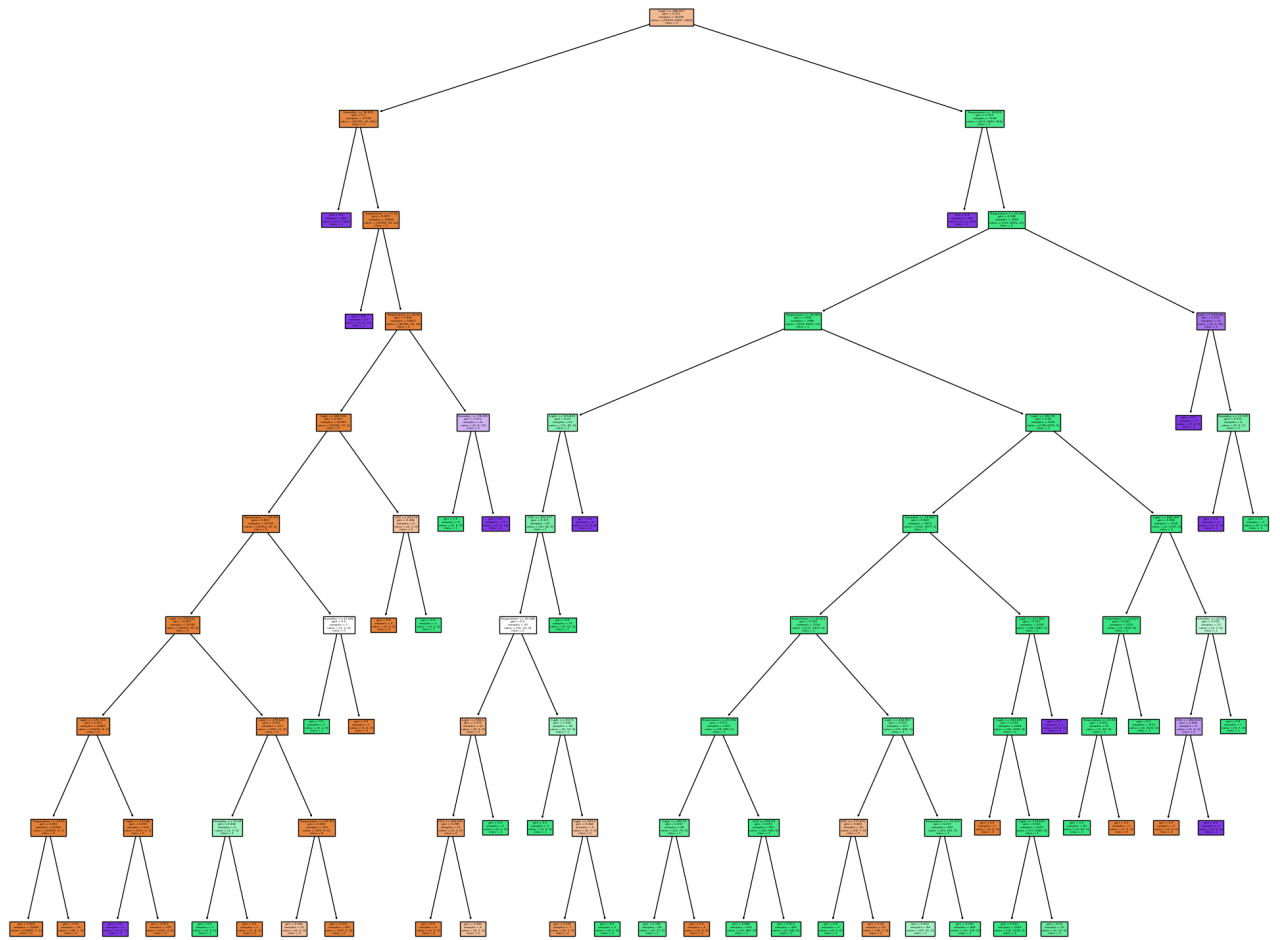


Figure 4.57: Decision Tree trained on the global dataset

- 1732 instances which are part of the class "Occupied" are correctly classified.
- 36 instances which are part of the class "Occupied" are wrongly classified as "Not Occupied".
- 3 instances are part of the class "Occupied" are correctly classified.
- 36 instances which are part of the class "Occupied" are wrongly classified as "disturbance".
- 338 instances which are part of the class "disturbance" are correctly classified.

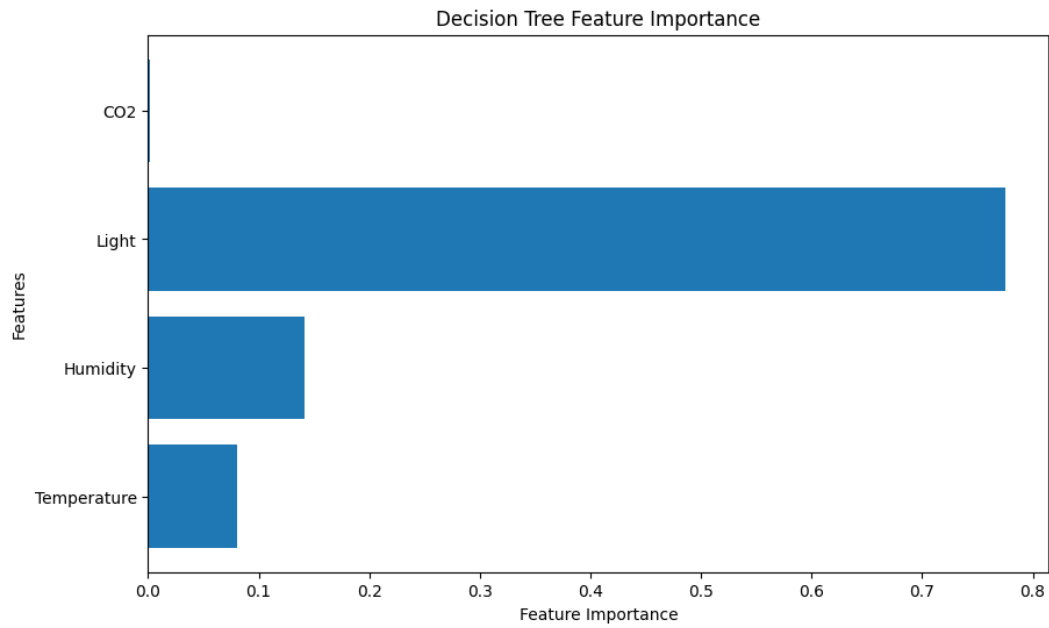


Figure 4.58: Feature Importance - Decision Tree trained on the global dataset

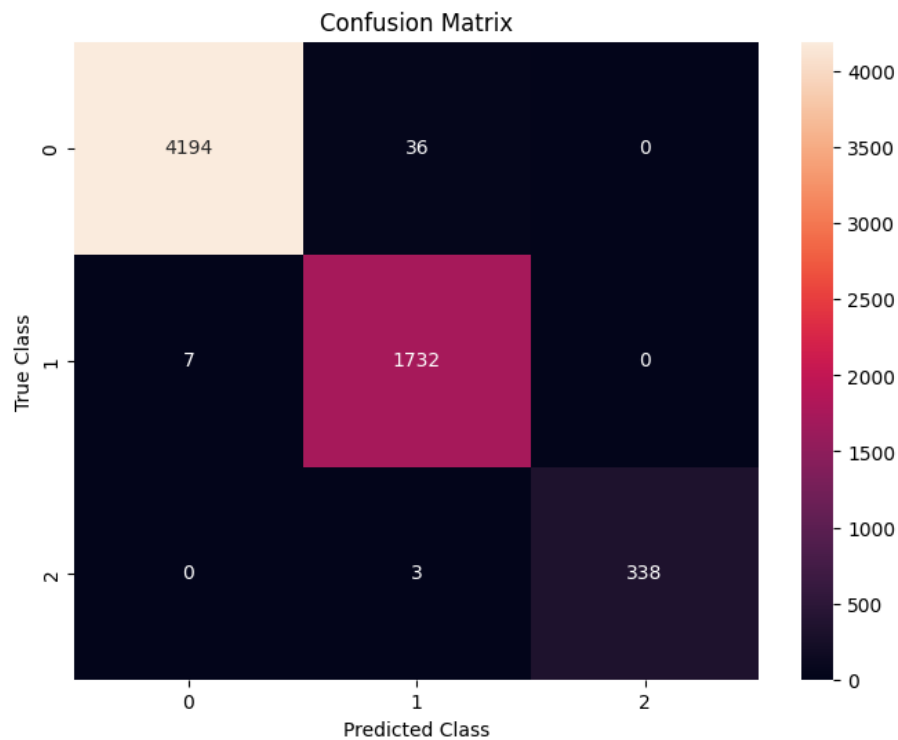


Figure 4.59: Confusion Matrix - Decision Tree trained on the global dataset

- No instances which are part of the class "disturbance" are wrongly classified as "Occupied".
- No instances which are part of the class "disturbance" are wrongly clas-

sified as "Not Occupied".

The scores resulted are the following:

- Accuracy: 99.27%
- Precision: 99.28%
- Recall: 99.27%
- F1\_Score: 99.27%

#### **5.3.2.2 Random Forest**

The random forest model performed similarly to the decision tree and gave performing results.

It gave importance to the Light feature (Figure 4.60

The scores resulted are the following:

- Accuracy: 99.38%
- Precision: 99.39%
- Recall: 99.38%
- F1\_Score: 99.38%

#### **5.3.2.3 KNN**

In Figure 4.61, different Number of Neighbors (K) are presented and each has a different error rate.

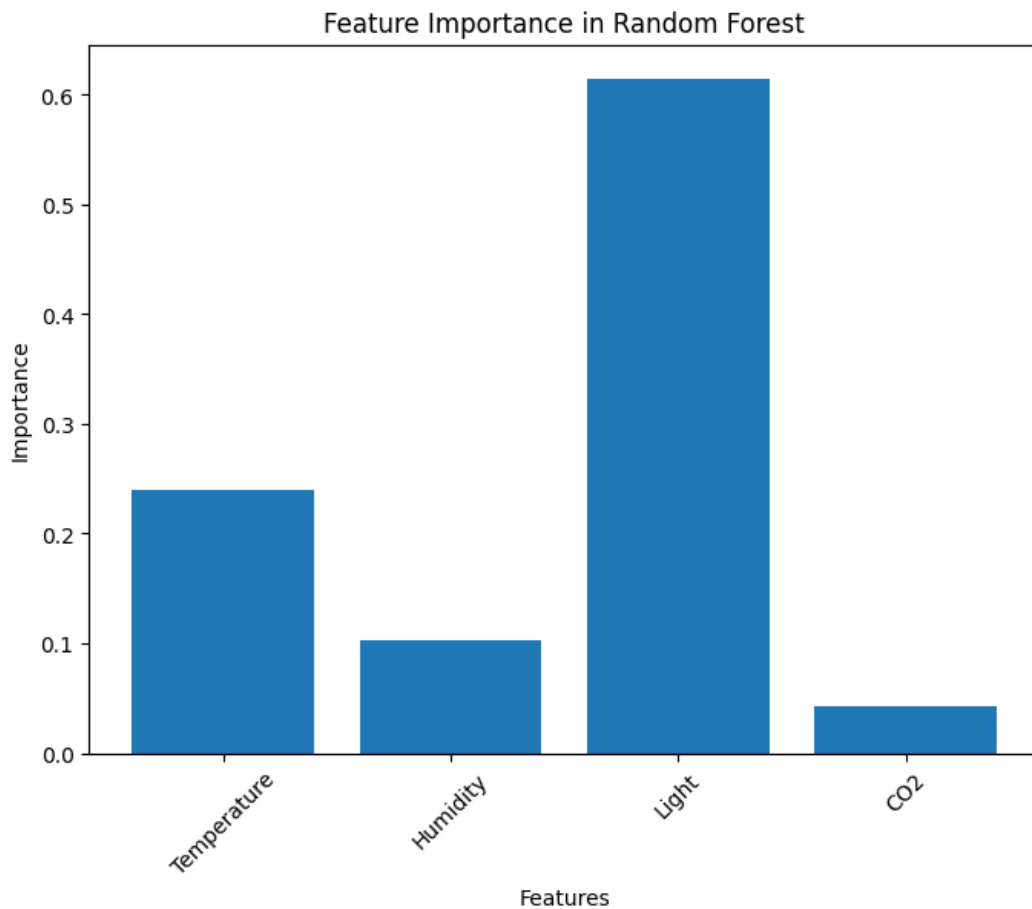


Figure 4.60: Feature Importance - Random Forest trained on the global dataset

And Figure 4.62 represents the Confusion Matrix which showcases the following results:

- 4225 instances which are part of the class "Not Occupied" are correctly classified.
- 21 instances which are part of the class "Not Occupied" are wrongly classified as "Occupied".
- 2 instances which are part of the class "Not Occupied" are wrongly classified as "disturbance".
- 1722 instances which are part of the class "Occupied" are correctly classified.
- 23 instances which are part of the class "Occupied" are wrongly classified

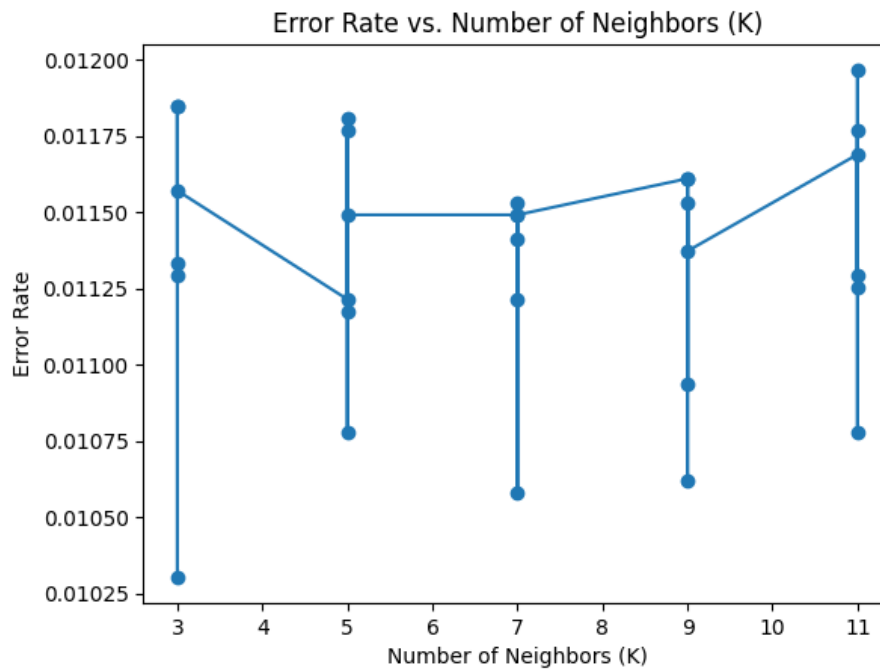


Figure 4.61: Number of Neighbors - Decision Tree trained on the global dataset

as "Not Occupied".

- 3 instances are part of the class "Occupied" are correctly classified.
- 313 instances which are part of the class "disturbance" are correctly classified.
- No instances which are part of the class "disturbance" are wrongly classified as "Occupied".
- Only One instance that is part of the class "disturbance" is wrongly classified as "Not Occupied".

The scores resulted are the following:

- Accuracy: 99.20%
- Precision: 99.20%
- Recall: 99.20%

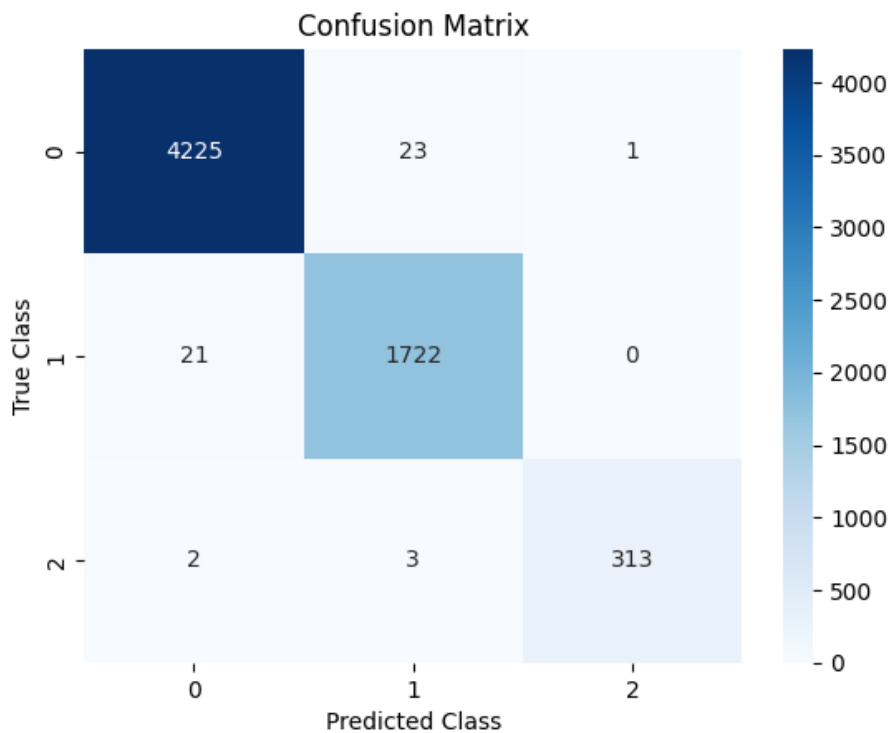


Figure 4.62: Confusion Matrix - KNN trained on the global dataset

- F1\_Score: 99.20%

#### 5.3.2.4 The DT Model Selection

Two choices are proposed here:

- Either pick the RNN model similar to the PT which is a model that takes into consideration the date yet it is not as performing as other models.
- Pick a model that is well performing and give well predictions mostly (Random Forest or Decision Tree) but i does not take into consideration the timestamp.

The model selected is the Decision Tree Model due to its performance on tabular data and the results it gave.

## 5.4 Languages and Libraries

- Languages
  - **Python:** a strong, high-level, general-purpose multi-platform programming language that is widely used in various purposes (AI, web development, etc).
- Libraries
  - **Paho Library:** an open-source MQTT client library that provides implementations in various programming languages, it allows the creation of MQTT clients and interact with MQTT brokers. The "paho.mqtt.publish" is used to publish messages to the MQTT broker and "paho.mqtt.client" is to subscribe to topics to receive the sent messages.
  - **Influxdb Library:** a Python client library that allows the interaction with the InfluxDB time-series database.
  - **hp206c Library:** provides a way of interaction with the HP206C sensor which is the barometer in the followed use case.
  - **Grovepi Library:** allow the interaction with the GrovePi+ board which means the interaction with any grove component (sensors, actuators etc.)
  - **Threading Library:** create and manage threads.
  - **Pandas Library:** a powerful open-source Python library used for data manipulation, analysis, and exploration. It provides data structures and functions to efficiently handle and process structured data
  - **Numpy Library:** manipulates matrices and performs mathematical operation.
  - **Matplotlib.pyplot:** to create plots and visualisation graphs.



- **Seaborn:** data visualization library built on top of matplotlib.
- **Jupyter Notebook:** an open-source web application that let the execution of sub-parts of Python code on one kernel and allows to create and share documents that combine live code and visualizations;

Chapter

5

# Demonstration

## Contents

1	The Physical Twin . . . . .	<b>151</b>
1.1	The first sub-system . . . . .	151
1.2	The second sub-system . . . . .	152
2	The Digital Twin . . . . .	<b>153</b>

# 1 The Physical Twin

The physical twin is composed of two sub-systems:

- **The first sub-system:** it consists of the system that collects the environmental data then use its corresponding model to make predictions. The results are sent to the second sub-system.
- **The second sub-system:** this second sub-system is the Poppy Ergo Jr and its OS. It receives the results from the first sub-system, and the robot moves accordingly.

## 1.1 The first sub-system

The first subsystem is in the form of an Edge/Fog.

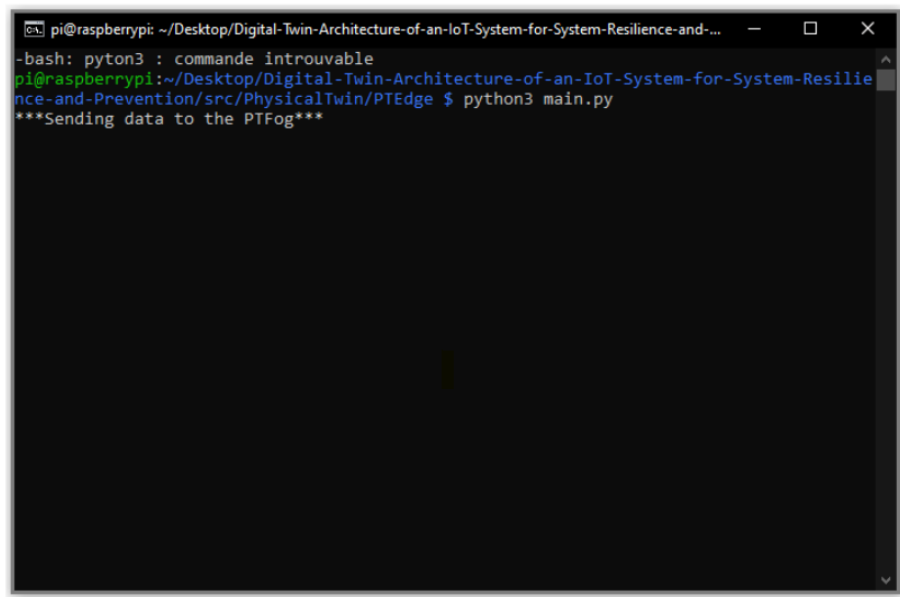
### 1.1.1 The first sub-system edge

As it can be seen in Figure 5.1, The edge is collecting the data (Temperature, Humidity, Light, CO2) and sending it directly to the fog.

### 1.1.2 The first sub-system fog

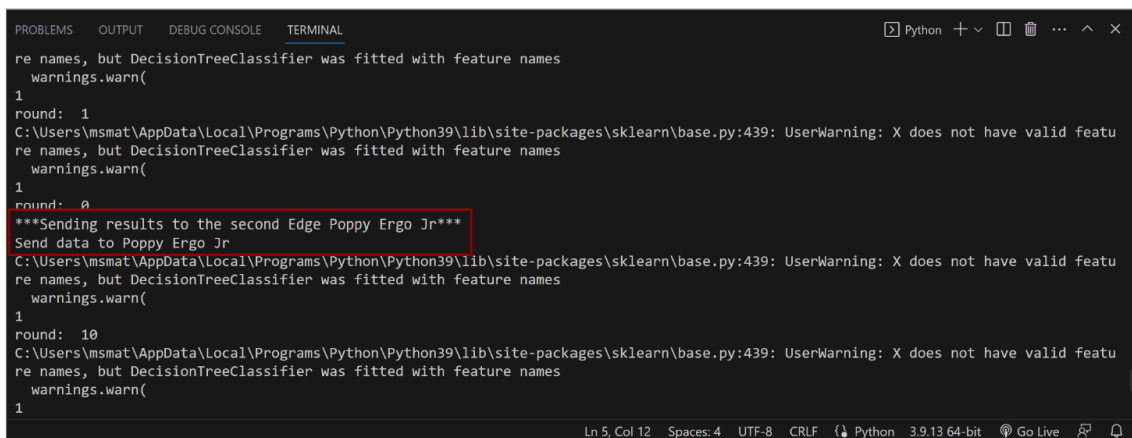
As the demonstration in Figure 5.2 shows, the fog receives the collected data from the fog and proceeds to use it as an input to the corresponding model.

At the end, the data and the result of the model are sent to the second edge of Poppy Ergo Jr.

A terminal window on a Raspberry Pi. The title bar shows the path: ~/Desktop/Digital-Twin-Architecture-of-an-IoT-System-for-System-Resilience-and-Prevention. The terminal shows the following commands and output:

```
pi@raspberrypi: ~/Desktop/Digital-Twin-Architecture-of-an-IoT-System-for-System-Resilience-and-Prevention/src/PhysicalTwin/PTEdge $ python3 main.py
***Sending data to the PTFog***
```

Figure 5.1: Demonstration - PTEdge

A VS Code terminal window showing the execution of a Python script. The terminal output includes several warnings from sklearn and a message indicating data is being sent to the second Edge Poppy Ergo Jr. The message is highlighted with a red box:

```
re names, but DecisionTreeClassifier was fitted with feature names
warnings.warn(
1
round: 1
C:\Users\msmat\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
warnings.warn(
1
round: 0
***Sending results to the second Edge Poppy Ergo Jr***
Send data to Poppy Ergo Jr
C:\Users\msmat\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
warnings.warn(
1
round: 10
C:\Users\msmat\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\base.py:439: UserWarning: X does not have valid feature names, but DecisionTreeClassifier was fitted with feature names
warnings.warn(
1
```

Figure 5.2: Demonstration - PTFog

## 1.2 The second sub-system

Figure 5.3 represents a web interface screenshot of the Poppy Ergo Jr OS . It showcases that the Poppy Ergo Jr do receive the data from the first sub-system fog and move accordingly.

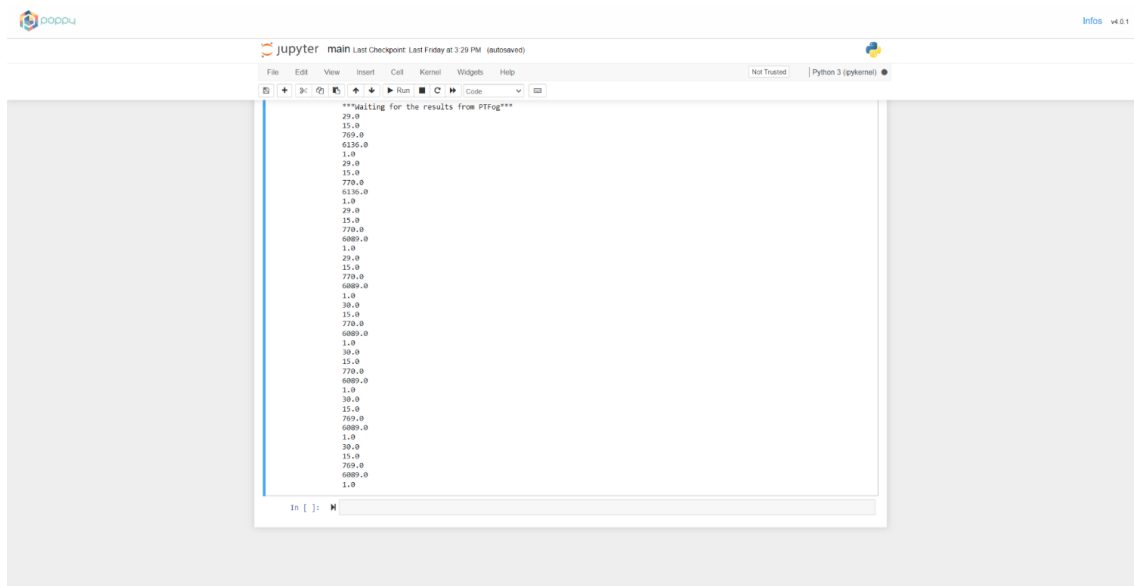


Figure 5.3: Demonstration - Poppy Ergo Jr

## 2 The Digital Twin

The Digital Twin is the replica of the first sub-system only since in the level of granularity, it is a system twin. Therefore, the Digital Twin is in the form of an Edge and a Fog (Figure 5.4)

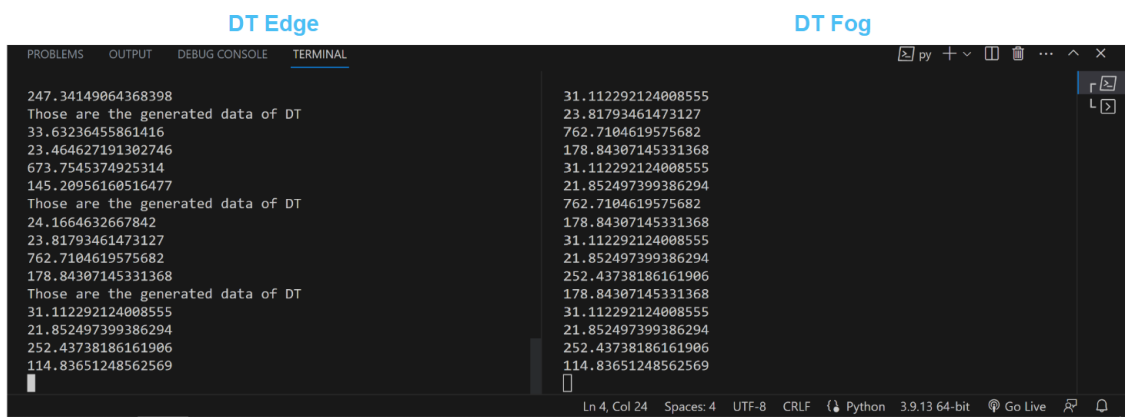


Figure 5.4: Demonstration - Digital Twin

- The DT Edge receives data from the PT and generate similar data depending on a configuration file, as well as disturbance. It then sends those

data to its fog.

- The fog receives the data from the DT edge and use it as an input to the corresponding model.

## Part V

# Prospective Endeavors and Synopsis

## Future Work

In this chapter, potential avenues for further exploration, development, and expansion of the current work are outlined.

The improvements, advancements and additional studies that will be undertaken to enhance the existing work are the following:

- Change the type of the used DT from a static twin to a dynamic one where data and responses are done in real-time.
- Change the level of granularity from a system twin to a process twin.
- Develop the use case and include reinforcement learning.
- Find a solution to include different environments in the use case to not have the problem of different distributed data.
- Add a cloud layer to upgrade the architecture to an Edge/Fog/Cloud and deal with big data.



## Part VI

### General Conclusion

## General Conclusion

In this Engineering degree report, an exploration of the power and potential of Digital Twins as a preventative and resilience tool was presented, along with relevant definitions and a comprehensive state-of-the-art analysis. However, it is crucial to acknowledge that while Digital Twins offer numerous benefits, they are not a one-size-fits-all solution for all maintenance and resilience challenges.

Implementing Digital Twins requires significant investments in terms of data collection, analytic capabilities, computing power, and the recruitment and training of skilled personnel. These resources are necessary for ensuring the proper operation and maintenance of Digital Twins. It is also important to note that while Digital Twins can provide valuable insights and analysis, they cannot entirely replace human intuition and expertise in decision-making processes. Instead, they should be viewed as a complementary tool that enhances and augments traditional maintenance practices.

That is why a cognitive super-Digital Twin has been implemented by testing different Machine Learning and Deep Learning methods and led to the creation of an advanced advanced version of the Digital Twin concept which not only

replicates the physical IoT Twin with precision but also possesses the unique capability to detect and generate perturbations for enhanced prevention and resilience strategies. By introducing controlled disruptions and analyzing their impact, the super-Digital Twin aims to improve the overall performance and preparedness of systems.

# Bibliography

- [1] I. Zhou, I. Makhdoum, N. Shariati, M. A. Raza, R. Keshavarz, J. Lipman, M. Abolhasan, and A. Amalipour, "Internet of things 2.0: Concepts, applications, and future directions," vol. 9, pp. 1–2, May 2021. [Online]. Available: <https://ieeexplore.ieee.org/document/9427249>
- [2] R. Minerva, A. Biru, and D. Rotondi, "Towards a definition of the internet of things (iot)," pp. 17–19, May 2015. [Online]. Available: [https://iot.ieee.org/images/files/pdf/IEEE\\_IoT\\_Towards\\_Definition\\_Internet\\_of\\_Things\\_Issue1\\_14MAY15.pdf](https://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Issue1_14MAY15.pdf)
- [3] K. K. Patel, S. M. Patel, and C. Salazar, "Internet of things-iot: Definition, characteristics, architecture, enabling technologies, application future challenges," vol. 6, p. 2, May 2016. [Online]. Available: [https://www.researchgate.net/publication/330425585\\_Internet\\_of\\_Things-IOT\\_Definition\\_Characteristics\\_Architecture\\_Enabling\\_Technologies\\_Application\\_Future\\_Challenges](https://www.researchgate.net/publication/330425585_Internet_of_Things-IOT_Definition_Characteristics_Architecture_Enabling_Technologies_Application_Future_Challenges)
- [4] F. I. Suny, M. Rahman, M. Roshed, and N. T. Newaz, "Iot past, present, and future a literary survey," p. 3, Jul. 2021. [Online]. Available: [https://www.researchgate.net/publication/353004403\\_IoT\\_Past\\_Present\\_and\\_Future\\_a\\_Literary\\_Surveys](https://www.researchgate.net/publication/353004403_IoT_Past_Present_and_Future_a_Literary_Surveys)

- [5] S. DuBravac, "The evolution of the internet of things (iot)," *Harvard Business Review*, vol. 93, pp. 72–75, Nov. 2015. [Online]. Available: <https://hbr.org/2015/11/the-evolution-of-the-internet-of-things-iot>
- [6] S. N. Swamy and S. R. Kota, "An empirical study on system level aspects of internet of things (iot)," vol. 8, pp. 19–20–21, Oct. 2020. [Online]. Available: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9218916>
- [7] (2021) Difference between cloud, fog and edge computing in iot. [Online]. Available: <https://www.digiteum.com/cloud-fog-edge-computing-iot/>
- [8] J. P. G. Sterbenz, "Smart city and iot resilience, survivability, and disruption tolerance: Challenges, modelling, and a survey of research opportunities," Nov. 2017. [Online]. Available: <https://ieeexplore.ieee.org/document/8093025>
- [9] H. P. R. J. D. F. J. H. G. H. Christian Berger, Philipp Eichhammer, "A survey on resilience in the iot: Taxonomy, classification and discussion of resilience mechanisms," Sep. 2021. [Online]. Available: <https://arxiv.org/abs/2109.02328>
- [10] IBM, "What is machine learning?" [Online]. Available: <https://www.ibm.com/topics/machine-learning>
- [11] E. Burns, "Machine learning," 2019. [Online]. Available: <https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML>
- [12] Javatpoint, "Machine learning algorithms." [Online]. Available: <https://www.javatpoint.com/machine-learning-algorithms>
- [13] J. point, "Unsupervised machine learning." [Online]. Available: <https://www.javatpoint.com/unsupervised-machine-learning>
- [14] S. Bhatt. Reinforcement learning 101. [Online]. Available: <https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292>

- [15] Wikipedia. Reinforcement learning. [Online]. Available: [https://en.wikipedia.org/wiki/Reinforcement\\_learning](https://en.wikipedia.org/wiki/Reinforcement_learning)
- [16] geeksforgeeks. Decision tree. [Online]. Available: <https://www.geeksforgeeks.org/decision-tree/>
- [17] T. Yiu. Understanding random forest. [Online]. Available: <https://towardsdatascience.com/understanding-random-forest-58381e0602d2/>
- [18] Wikipedia. Naive bayes classifier. [Online]. Available: [https://en.wikipedia.org/wiki/Naive\\_Bayes\\_classifier](https://en.wikipedia.org/wiki/Naive_Bayes_classifier)
- [19] (Jun.) Convolutional neural network : Tout ce qu'il y a à savoir. [Online]. Available: <https://datascientest.com/convolutional-neural-network>
- [20] A. A. A. M. S. J. E. T. Chukwudi Nwogu, Giovanni Lugaresi. (2020) The digital twin paradigm for smarter systems and environments: The industry use cases. [Online]. Available: <https://www.sciencedirect.com/topics/computer-science/multilayer-perceptron>
- [21] A. Biswal. (2023, Apr.) Recurrent neural network(rnn) tutorial: Types, examples, lstm and more. [Online]. Available: <https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn>
- [22] M. Saeed. (2022, Sep.) An introduction to recurrent neural networks and the math that powers them. [Online]. Available: <https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/>
- [23] A. Chugh. (2021) Deep learning | introduction to long short term memory. [Online]. Available: <https://www.geeksforgeeks.org/deep-learning-introduction-to-long-short-term-memory/>
- [24] Q. H. M. Imtiaz Ullah. (2022, Jun.) Design and development of rnn anomaly detection model for iot net-

- works. [Online]. Available: <https://machinelearningmastery.com/an-introduction-to-recurrent-neural-networks-and-the-math-that-powers-them/>
- [25] Javatpoint. Difference between machine learning and deep learning. [Online]. Available: <https://www.javatpoint.com/machine-learning-vs-deep-learning>
- [26] K. Lamb, “Principle-based digital twins: a scoping review,” pp. 19–20–21, Dec. 2019. [Online]. Available: [https://www.cdbb.cam.ac.uk/files/scopingreview\\_dec20.pdf](https://www.cdbb.cam.ac.uk/files/scopingreview_dec20.pdf)
- [27] X. D. Z. L. J. T. Weifei Hu, Tongzhou Zhang, “Digital twin: a state-of-the-art review of its enabling technologies, applications and challenges,” *Journal of Intelligent Manufacturing and Special Equipment*, pp. 19–20–21, Jul. 2021. [Online]. Available: <https://www.emerald.com/insight/content/doi/10.1108/JIMSE-12-2020-010/full/html>
- [28] A. Sharma, E. Kosasih, J. Zhang, A. Brintrup, and A. Calinescu, “Digital twins: State of the art theory and practice, challenges, and open research questions,” pp. 19–20–21, Dec. 2020. [Online]. Available: <https://arxiv.org/pdf/2011.02833.pdf>
- [29] E. H. Glaessgen and D. S. Stargel, “The digital twin paradigm for future nasa and u.s. air force vehicles,” p. 7, 2012. [Online]. Available: <https://ntrs.nasa.gov/api/citations/20120008178/downloads/20120008178.pdf>
- [30] J. D. Hochhalter, W. P. Lester, J. A. Newman, E. H. Glaessgen, V. K. Gupta, V. Yamakov, S. R. Cornell, S. A. Willard, and G. Heber, “Coupling damage-sensing particles to the digital twin concept,” p. 5, Apr. 2014. [Online]. Available: <https://ntrs.nasa.gov/api/citations/20140006408/downloads/20140006408.pdf>
- [31] A. Parrott and L. Warshaw, “Industry 4.0 and the digital twin,” 2021. [Online]. Available: <https://www2.deloitte.com/us/en/insights/>

<focus/industry-4-0/digital-twin-technology-smart-factory.html>

- [32] F. Tao, M. Zhang, Y. Liu, and A. Y. C. Nee, “Digital twin driven prognostics and health management for complex equipment,” vol. 67, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/abs/pii/S0007850618300799>
- [33] M. N. K. Boulos and P. Zhang, “Digital twins: From personalised medicine to precision public healths,” Jul. 2021. [Online]. Available: <https://www.mdpi.com/2075-4426/11/8/745>
- [34] W. Purcell, T. Neubauer, and K. Mallinger, “Digital twins in agriculture: challenges and opportunities for environmental sustainability,” *Current Opinion in Environmental Sustainability*, vol. 61, p. 1, Apr. 2023. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S187734352200104X>
- [35] P. Brosset, T. Matins, M. Watson, and S. Williams, “How digital twins enable autonomous operations,” Feb. 2022. [Online]. Available: <https://www.accenture.com/us-en/insights/industry-x/manufacturing-systems-architecture>
- [36] M. Grieves and J. Vickers, “Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems,” p. 3, Aug. 2017. [Online]. Available: [https://www.researchgate.net/publication/306223791\\_Digital\\_Twin\\_Mitigating\\_Unpredictable\\_Undesirable\\_Emergent\\_Behavior\\_in\\_Complex\\_Systems](https://www.researchgate.net/publication/306223791_Digital_Twin_Mitigating_Unpredictable_Undesirable_Emergent_Behavior_in_Complex_Systems)
- [37] M. Grieves, “Digital twin: Manufacturing excellence through virtual factory replication,” p. 3, Mar. 2015. [Online]. Available: [https://www.researchgate.net/publication/275211047\\_Digital\\_Twin\\_Manufacturing\\_Excellence\\_through\\_Virtual\\_Factory\\_Replication](https://www.researchgate.net/publication/275211047_Digital_Twin_Manufacturing_Excellence_through_Virtual_Factory_Replication)



- [38] A. Badach, “Digital twins in iot,” pp. 19–20–21, Dec. 2022. [Online]. Available: [https://www.researchgate.net/publication/366167747\\_Digital\\_Twins\\_in\\_IoT](https://www.researchgate.net/publication/366167747_Digital_Twins_in_IoT)
- [39] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, “Internet of things 2.0: Concepts, applications, and future directions,” p. 8, Mar. 2020. [Online]. Available: [https://www.researchgate.net/publication/339802823\\_Characterising\\_the\\_Digital\\_Twin\\_A\\_systematic\\_literature\\_review](https://www.researchgate.net/publication/339802823_Characterising_the_Digital_Twin_A_systematic_literature_review)
- [40] J. Shepard, “What’s a digital shadow and how does it relate to a digital twin?” Sep. 2022. [Online]. Available: <https://www.analogictips.com/whats-a-digital-shadow-and-how-does-it-relate-to-a-digital-twin-faq/>
- [41] W. SA, “Difference between digital twin, digital model, and digital shadow.” [Online]. Available: <https://www.wizata.com/knowledge-base/difference-between-digital-twin-digital-model-and-digital-shadow>
- [42] C. W. Wil Van der Aalst, Oliver Hinz, “Resilient digital twins: Organizations need to prepare for the unexpected,” Sep. 2021. [Online]. Available: [https://www.researchgate.net/publication/354764507\\_Resilient\\_Digital\\_Twins\\_Organizations\\_Need\\_to\\_Prepere\\_for\\_the\\_Unexpected](https://www.researchgate.net/publication/354764507_Resilient_Digital_Twins_Organizations_Need_to_Prepere_for_the_Unexpected)
- [43] L. Seppälä, “Digital model, digital shadow, or digital twin – what is at the core of data-driven shipbuilding?” pp. 19–20–21, Aug. 2020. [Online]. Available: <https://www.cadmatic.com/en/resources/blog/digital-model-digital-shadow-or-digital-twin/>
- [44] T. A. A. K. D. S. T. A. Thomas Bergs, Sascha Gierlings, “The concept of digital twin and digital shadow in manufacturing,” vol. 101, Sep. 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2212827121006612>

- [45] G. Fuller, “Digital model, digital shadow, and digital twin: their places in eiot,” Sep. 2021. [Online]. Available: <https://www.psa.inc/company/news/digital-model-vs-digital-shadow-vs-digital-twin-and-their-place-in-eiot/>
- [46] D. D. Teams, “4 types of digital twins (basic overview with examples),” Nov. 2022. [Online]. Available: <https://digitaldirections.com/4-types-of-digital-twins-basic-overview-with-examples/#what-are-the-main-types-of-digital-twins>
- [47] IBM, “What is a digital twin?” Jun. 2022. [Online]. Available: <https://www.ibm.com/topics/what-is-a-digital-twin>
- [48] M. Kor, “Integration of digital twin and deep learning for facilitating smart planning and construction: An exploratory analysis,” Jun. 2021. [Online]. Available: <https://hj.diva-portal.org/smash/get/diva2:1569258/FULLTEXT01.pdf>
- [49] S. Software, “Machine learning supercharges real-time digital twins,” May 2021. [Online]. Available: <https://www.scaleoutsoftware.com/featured/machine-learning-supercharges-real-time-digital-twins/>
- [50] S. A. Mergen Kor, Ibrahim Yitmen, “An investigation for integration of deep learning and digital twins towards construction 4.0,” Mar. 2022. [Online]. Available: <https://www.emerald.com/insight/content/doi/10.1108/SASBE-08-2021-0148/full/html>
- [51] P. Sarkar, “Digital twin modeling using machine learning and constrained optimization,” Oct. 2022. [Online]. Available: <https://towardsdatascience.com>
- [52] R. Dolphin, “Lstm networks | a detailed explanation,” Oct. 2021. [Online]. Available: <https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9>

- [53] L. Grinsztajn, E. Oyallon, and G. Varoquaux, “Why do tree-based models still outperform deep learning on tabular data?” 2022. [Online]. Available: <https://arxiv.org/abs/2207.08815>
- [54] F. Flammini, “Digital twins as run-time predictive models for the resilience of cyber-physical systems: a conceptual framework,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 379, no. 2207, p. 20200369, 2021. [Online]. Available: <https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2020.0369>
- [55] P. Eirinakis, S. Lounis, S. Plitsos, G. Arampatzis, K. Kalaboukas, K. Kenda, J. Lu, J. M. Rožanec, and N. Stojanovic, “Cognitive digital twins for resilience in production: A conceptual framework,” *Information*, vol. 13, no. 1, 2022. [Online]. Available: <https://www.mdpi.com/2078-2489/13/1/33>
- [56] v. d. H. J. Bruynseels K, Santoni de Sio F, “Digital twins in health care: Ethical implications of an emerging engineering paradigm,” Feb. 2018. [Online]. Available: <https://pubmed.ncbi.nlm.nih.gov/29487613/>
- [57] T. Mortlock, D. Muthirayan, S. Y. Yu, P. P. Khargonekar, and M. A. A. Faruque, “Graph learning for cognitive digital twins in manufacturing systems,” *IEEE Transactions on Emerging Topics in Computing*, vol. 10, pp. 34–45, 2021.
- [58] C. Gao, H. Park, and A. Easwaran, “An anomaly detection framework for digital twin driven cyber-physical systems,” in *Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems*, ser. ICCPS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 44–54. [Online]. Available: <https://doi.org/10.1145/3450267.3450533>

- [59] S. R. Chhetri, S. Faezi, A. Canedo, and M. A. A. Faruque, “Quilt: Quality inference from living digital twins in iot-enabled manufacturing systems,” in *Proceedings of the International Conference on Internet of Things Design and Implementation*, ser. IoTDI '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 237–248. [Online]. Available: <https://doi.org/10.1145/3302505.3310085>
- [60] R. Pi. Raspberry pi 3 model b+. [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/>
- [61] Tutorial45. Best operating systems for raspberry pi os. [Online]. Available: <https://tutorial45.com/best-operating-systems-for-raspberry-pi-os/>
- [62] Seeed Studio. (Accessed June 2, 2023) GrovePi. [Online]. Available: <https://www.seeedstudio.com/GrovePi.html>
- [63] S. Studio. (Year not specified) Grove - temperature humidity sensor. [Online]. Available: [https://wiki.seeedstudio.com/Grove-TemperatureAndHumidity\\_Sensor/](https://wiki.seeedstudio.com/Grove-TemperatureAndHumidity_Sensor/)
- [64] ——. (Year not specified) Grove - barometer high accuracy. [Online]. Available: <https://wiki.seeedstudio.com/Grove-Barometer-High-Accuracy/>
- [65] ——. Grove - light sensor. [Online]. Available: [https://wiki.seeedstudio.com/Grove-Light\\_Sensor/](https://wiki.seeedstudio.com/Grove-Light_Sensor/)
- [66] ——. Grove - voc and eco2 gas sensor - sgp30. [Online]. Available: [https://wiki.seeedstudio.com/Grove-VOC\\_and\\_eCO2\\_Gas\\_Sensor-SGP30/](https://wiki.seeedstudio.com/Grove-VOC_and_eCO2_Gas_Sensor-SGP30/)
- [67] RabbitMQ. Rabbitmq. [Online]. Available: <https://rabbitmq.com/>
- [68] ——. Rabbitmq mqtt. [Online]. Available: <https://www.rabbitmq.com/mqtt.html>
- [69] Wikipedia contributors. Influxdb. [Online]. Available: <https://en.wikipedia.org/wiki/InfluxDB>

- [70] P. Project. Poppy ergo jr. [Online]. Available: <https://www.poppy-project.org/en/robots/poppy-ergo-jr/>
- [71] ——. Poppy project documentation. [Online]. Available: <https://docs.poppy-project.org/en/>
- [72] A. A. A. M. S. J. T. Chukwudi Nwogu, Giovanni Lugaresi, “Towards a requirement-driven digital twin architecture,” vol. 107, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2212827122003420>
- [73] A. S. G. TechTarget. (2022, Jun.) System of systems (sos). [Online]. Available: <https://www.techtarget.com/searchapparchitecture/definition/system-of-systems-SoS>
- [74] Wikipedia contributors. (2023, May) System of systems. [Online]. Available: [https://en.wikipedia.org/wiki/System\\_of\\_systems](https://en.wikipedia.org/wiki/System_of_systems)